UNIVERSITE DE GENEVE
Faculté de Psychologie
et des Sciences de l'Education
TECFA

Construire des Interactions au moyen d'Authorware 6

Version 4 (novembre 2002)

Pierre Dillenbourg & Patrick Jermann & Elia De Iaco

TECFA

http://tecfa.unige.ch/guides/authorware/

Table des matières

Leçon 1. Les icônes de base 7

1-1 Objectif 7
1-2 Introduction programmation graphique 7
1-3 Mode auteur et mode utilisateur 8
1-4 Mode vectoriel et mode ‘calque’ 9
1-5 Paramètres de l’icône d’effacement. 10
1-6 Paramètres de l’icône de temporisation (icône “WAIT”) 11
1-7 Organiser son programme 12

Leçon 2. Construire une interaction simple 13

2-1 Objectif 13
2-2 Modalités d’interaction 13
2-3 Structure d’une interaction simple 13
2-4 Gestion d’écran 17
2-5 Compter les bonnes et mauvaises réponses 18
2-6 Autres options du mode de réponse ‘bouton’ 19
2-7 Biais pédagogique dans la terminologie 20
2-8 Types de bouton et librairie de boutons 21
2-9 Comment créer de nouveaux boutons 21

Leçon 3. La manipulation directe 22

3-1 Objectif 22
3-2 Les micro-interactions 22
3-3 Les réponses de type 'zone sensible' 23
3-4 Gestion d'écran 24
3-5 Les réponses de type 'objet sensible' 25
3-6 Les réponses de type 'objet mobile' 26
3-7 Librairie de curseurs 27

Leçon 4. Structure des interactions 28

4-1 Objectif 28
4-2 Limiter le nombre d'essais 28
4-3 Limiter le temps de réponse 28
4-4 Analyse de réponse: comprendre l'ordre des tests 29
4-5 Analyse de réponse: une recette 31

Leçon 5. Les réponses 'texte' 33

5-1 Objectif 33
5-2 Principes de base 33
5-3 Paramétrage de l'analyseur 34
5-4 Définition du pattern 35

5-5 Feedback d'erreur générique 36
5-6 Patterns conjonctifs 36
5-7 Patterns disjonctifs 37
5-8 Patterns complexes 38
5-9 Analyses négatives 39
5-10 Traitement du langage naturel 40
5-11 Micro-interactions 40
5-12 Réponses de type 'touche' 42

Leçon 6. Variables et fonctions 43

6-1 Objectif 43
6-2 Variables Authorware 43
6-3 Les fonctions offertes par Authorware 44
6-4 Créer vos propres variables 47
6-5 Format de présentation des nombres 47
6-6 Structure itérative 48
6-7 Structure conditionnelle 49
6-8 Algorithmique 51
6-9 Commentaires 52

Leçon 7. Intégration des composantes multimédia 53

7-1 Objectif 53
7-2 Philosophie d'Authorware en matière de multimédia 53
7-3 L'icône son 53
7-4 L'icône vidéo 54
7-5 L'icône film 55
7-6 Animations à destination fixe 55
7-7 Animations à destination variable 57
7-8 Animations dans le plan 58
7-9 Positionnement variable sans animation 58
7-10 Déplacer un film 59
7-11 Librairies 59

Leçon 8. Structuration du logiciel 61

8-1 Objectif 61
8-2 Créer une boucle composée d'icônes 61
8-3 Tracer le fonctionnement d'Authorware 61
8-4 Sélectionner des icônes 62
8-5 Time limits 63
8-6 La fonction aléatoire 63
8-7 Les fonctions 'caractères' 64
8-8 Icônes ou fonctions? 65
8-9 Les modèles 65

Leçon 9. Interactions complexes 67

9-1 Objectif 67
9-2 Traiter les réponses de type 'texte' 67
9-3 Traitement des réponses numériques 67
9-4 Traitement des réponses à une question comprenant des variables 69
9-5 Traitement des réponses multiples 69

Leçon 10. Interactions perpétuelles 72

10-1 Objectif 72
10-2 Les réponses perpétuelles de type 'bouton' 72
10-3 Les réponses perpétuelles de type 'menu' 73
10-4 Réponses conditionnelles perpétuelles 74
10-5 Autres réponses perpétuelles 74
10-6 Gestion du temps 74
10-7 Hygiène de programmation 75

Leçon 11. La navigation dans les hyperdocuments 76

11-1 Objectif 76
11-2 Créer la structure d'un hypertexte: icône framework 76
11-3 Gérer la navigation 78
11-4 Créer des mots-sensibles 79
11-5 Gestion des hyperdocuments 80
11-6 Créer des schémas contenant des boutons de l'hypertexte 80

Leçon 12. La gestion des fichiers 81

12-1 Objectif 81
12-2 Les tableaux de données 81
12-3 Lire et analyser un fichier externe 84
12-4 Sauver les données 86

13. Aide 88

Introduction

Pourquoi avoir choisi Authorware? Il s'agit d'un langage de haut niveau spécialisé pour la conception de logiciels éducatifs, dans la lignée des langages Tutor, Pilot et autre TenCORE. Authorware est plus spécialisé que des systèmes tels que Visual Basic ou Toolbook qui n'ont pas été conçus spécifiquement pour développer des applications éducatives. Il se différencie aussi de produits tels que Director qui permettent de créer des présentations: ces derniers sont plus performants sur le plan des effets visuels et sonores (surtout les animations 3D), mais sont moins riches sur le plan de l'interaction.

Ce cours n'a pas pour vocation de se substituer au mode d'emploi livré avec le logiciel. Nous ne décrivons par exemple pas les menus de façon systématique, nous ne nous concentrons pas sur Authorware per se, mais sur ce qu'on peut faire avec Authorware. Les modes d'emplois sont disponibles à la bibliothèque de la Faculté. (FPSE).

Authorware fonctionne sur Mac et sous Windows et possède une interface auteur qui est pratiquement identique sur les deux machines. Toutefois, ce cours adopte la politique de l'Université de Genève, à savoir se concentrer sur la plateforme Window. Nous travaillons avec la version 6 de Authorware. Vous pouvez travailler avec la version 4 pour Mac, mais le matériel de cours n'est plus adapté. Quoi de neuf dans la version 6 ? Sa logique de fonctionnement est semblable à la version 3.5 utilisée jusque récemment dans ce cours. Les principales nouveautés sont:

- Une réorganisation de l'interface, davantage 'windows', notamment de la façon d'éditer des icônes;
- Un outil d'édition de haut niveau qui permet d'importer directement des logiciels-type, par exemple d'ouvrir et éditer un QCM. Ces 'knowledge objects' ne sont pas décrits dans cette version du cours.
- Un enrichissement de diverses fonctions, notamment en ce qui concerne la gestion de la formation (fonctions de Computer-Managed Instruction), de gestion des données et d'interface vers le Web.

Ces nouveautés n'ont pas encore été toutes intégrées dans le présent matériel de formation.

Bien que ce document concerne uniquement Authroware, nous plaçons déjà en note de bas de pages quelques considérations ergonomiques qui seront reprises dans la partie théorique du cours 'Interaction Personne-Machine'.

Mode d'emploi

Ces notes de cours sont un complément aux activités prévues dans le Practicum Authorware atteignable depuis la page du cours Interaction Personne-Machine. Vous y trouverez, leçon par leçon (salle par salle):

- Les **énoncés** des exercices, accompagnés d'une aide.

- Les **corrigés**, c-à-d les programmes Authorware correspondant à l'énoncé, celui créé par l'enseignant et ceux créés par d'autres étudiants. Ces corrigés sont accessibles directement depuis le campus virtuel . Il s'agit de fichiers Authorware que l'on télécharge ('enregister sous...') avec Netscape en cliquant sur le lien hypertexte avec le bouton droit de la souris (sur MAC, cliquer en pressant la touche Control). En réalité, il n'y a pas de 'bonne' solution. Sauf exception, si votre programme fait ce qui est décrit dans l'énoncé, c'est bon. Les corrigés sont plutôt utiles si vous ne voyez pas comment vous y prendre et cherchez un modèle.

- Si vous avez une question, postez-la sur le **forum de discussion** (news://tecfa2.unige.ch/topics.authorware). Quel est l'avantage d'un tel système par rapport au courrier électronique traditionnel? Tout d'abord, il vous permet de voir si votre question a déjà été posée et, le cas échéant, si elle a déjà reçu une réponse. Ensuite, vous pouvez apprendre en voyant les questions de vos collègues qui soulèvent peut-être un point auquel vous n'aviez pas pensé. Finalement, vous aurez l'occasion de répondre vous-même à des questions dont vous connaissez la réponse.

Copyright

Ce cours est disponible en libre accès à tout particulier, y compris à ceux qui ne sont pas inscrits à l'un des cours de TECFA. Par contre, ce cours n'est pas mis à la disposition des institutions désirant l'utiliser pour leur cours. Ces institutions sont priées de contacter les auteurs pour utiliser ce matériel.

Leçon 1. Les icônes de base

1-1 Objectif

Dans cette première leçon, vous apprendrez à créer un programme très simple, lequel présente des informations pendant un certain temps, puis les efface.

1-2 Introduction programmation graphique

Construire un logiciel dans Authorware ne consiste pas à rédiger des lignes de code comme dans un langage classique, mais à dessiner l'ordinogramme du logiciel en assemblant des icônes. Cela se passe en trois temps:

1. Authorware présente une palette comprenant 13 icônes. Ces icônes représentent différents éléments de programme permettant d'afficher de l'information, de poser une question, d'animer un objet, etc...
2. Dans la fenêtre programme se trouve une barre verticale appelée ci-après **barre de déroulement** (en anglais '*flowline***'**). Cette ligne représente le déroulement chronologique du programme. **Construire un programme consiste à prélever des icônes dans une palette et à les déposer sur la barre de déroulement**: il suffit de cliquer sur une icône dans la palette et de la tirer vers la barre de déroulement. Authorware exécute séquentiellement du haut vers le bas toutes les icônes placées sur la barre de déroulement.
3. Ensuite, il faut éditer chaque icône pour en préciser le contenu ou en modifier certains paramètres. Les paramètres varient selon l'icône. Nous étudierons ces icônes une par une tout au long de ce cours.

L'auteur ne dessine donc pas manuellement l'ordinogramme. Les traits entre les icônes sont générés par Authorware selon l'endroit où l'icône a été déposée et selon les valeurs de paramètres fixés par l'auteur lorsqu'il édite ces icônes.

Avant de continuer, il est préférable que vous fassiez l'exercice 1.1. qui vous explique comment construire votre premier programme. Comme il s'agit de votre première leçon, nous vous décrivons pas à pas comment procéder.

Exercice 1-1: Votre premier programme

Premier pas: créer un programme vide

Quand vous lancez l'application Authorware, celui-ci affiche une boîte de dialogue afin de vous permettre de sélectionner un fichier à ouvrir[1]. Au lieu de choisir un fichier parmi ceux qui vous sont proposés, cliquez sur le bouton ***none****. Authorware crée alors une fenêtre vide, comprenant une barre verticale[2]. Cette fenêtre s'appelle* ***Untitled.*** *C'est le nom provisoire de votre programme. Authorware ouvre également une fenêtre 'Kowledge Objects' qui ne vous est pas utile actuellement, il est préférable de la fermer.*

Deuxième pas: dessiner l'écran que verra l'utilisateur

La première icône en haut de la palette est ***l'icône de présentation****. Faire un 'drag and drop' de cette icône vers votre programme. Authorware place en haut du programme une copie de l'icône et la dénomme 'untitled'[3]. Double-cliquez sur cette icône et la fenêtre qui apparaît*

1. Si Authorware est déjà lancé, vous pouvez créer un nouveau programme en sélectionnant l'option 'new' dans le menu 'file'.
2. Authorware ne permet de travailler que sur un programme à la fois. Si vous demandez 'open' ou 'new' alors qu'un programme est déjà ouvert, Authorware vous demandera de fermer d'abord le programme en cours d'édition ou fermera celui-ci si aucun changement n'a été effectué depuis la dernière sauvegarde
3. Vous pouvez modifier le nom de l'icône. Quand un programme comporte de nombreuses icônes, il est essentiel de donner un nom différent à chaque icône afin d'augmenter la lisibilité du programme.

est exactement celle que verra l'utilisateur. Dessinez un cercle rouge dans le coin supérieur droit et l'utilisateur verra exactement ce cercle rouge à cet endroit ![4]*. Lors de l'édition d'un écran, une palette apparaît comportant 8 outils graphiques permettant de dessiner cet écran: taper du texte, dessiner des lignes, rectangles, ellipses, etc...*[5]*. Elle constitue une version réduite de ce qu'offrent les logiciels de dessin. Vous trouverez en outre dans le menu 'Window/ Inspectors' les options 'lines' , 'fills', 'modes' et 'color' qui permettent d'éditer les objets de l'écran, et dans le menu text, les options habituelles d'édition de texte (polices, styles, taille et alignement)*[6]*. Ces fonctionnalités sont relativement standard et documentées dans le guide de l'utilisateur. Certaines d'entre elles seront étudiées ultérieurement, telles que la possibilité de positionner un objet de façon variable ou d'intégrer la valeur d'une variable dans un texte.*

Troisième pas: dire au programme d'effacer l'écran

Prenez dans la palette l'icône d'effacement, symbolisée par une gomme (la troisième depuis le haut) et la placer sous la première icône de votre programme. Si vous cliquez deux fois sur cette icône, une petite fenêtre apparaît avec différents paramètres que nous étudierons plus tard. Il faut simplement dire à Authorware ce qu'il faut effacer. Pour cela, cliquez sur le cercle rouge que vous avez créé dans l'icône de présentation et qui apparaît en principe dans la fenêtre de présentation. Authorware efface ce cercle (ainsi que tous les autres objets que vous auriez éventuellement ajoutés à cette icône). Authorware vous confirme qu'il a bien identifié ce qu'il fallait effacer en indiquant dans la boite de dialogue le nom de l'icône dont il a effacé le contenu (probablement qu'à ce stade cette icône s'appelle toujours 'Untitled').

Quatrième pas: exécuter votre programme

En fait, vous venez de créer votre premier programme. Vous pouvez maintenant le faire tourner en choisissant l'option 'restart' dans le menu 'control'. Vous verrez le cercle rouge apparaître puis s'effacer immédiatement. Voilà, votre programme est fini. L'utilisateur n'aura donc pas eu le temps de voir l'information que vous lui présentez. Pour améliorer cela, ajoutons une temporisation entre l'affichage et l'effacement.

Cinquième pas: insérer une pause

Prenez dans la palette, l'icône 'wait' (la quatrième depuis le haut) et placez-la entre l'icône de présentation et l'icône d'effacement. Il n'est pas nécessaire d'éditer les paramètres de cette icône pour le moment. Exécutez à nouveau le programme: celui-ci affiche le contenu de l'icône d'interaction, puis s'interrompt tant que l'utilisateur n'a pas poussé sur le bouton 'continue'. Dès que ce bouton est pressé, le programme reprend son fil, c'est-à-dire exécute l'icône suivante sur la barre de déroulement, l'effacement.

Bravo, vous êtes au bout de votre premier programme Authorware !

1-3 Mode auteur et mode utilisateur

Au cours de l'exercice 1.1, tantôt vous étiez le concepteur d'un logiciel et tantôt, vous étiez dans la peau de celui qui utilisera ultérieurement ce programme. Il existe donc deux points de vue sur ce programme: celui de l'auteur et celui de l'utilisateur, respectivement liés à l'édition et à l'exécution du programme. Un des grands point forts d'Authorware réside dans la facilité de passer d'un mode à l'autre à n'importe quel moment. Cependant, cet avantage déroute parfois le programmeur débutant. En cas de doute, sachez que si les

4. Dans un langage moins évolué, il faudrait décrire l'écran plutôt que de le dessiner, en entrant une commande du genre " drawcircle (topwindow, red, 214,4,233,65) ". La possibilité de dessiner l'écran tel qu'il apparaît permet une économie importante sur la mise au point de ces écrans. Elle entraîne toutefois certains inconvénients liés à toute forme de manipulation directe. Cette méthode ne permet par exemple pas de dessiner un cercle dont la taille est déterminée en cours d'exécution du programme (par exemple, le cercle rétrécit lorsque les réserves d'oxygène du vaisseau spatial s'épuisent). Toutefois, nous verrons qu'Authorware supporte partiellement cette approche (utilisation de fonctions de dessin dans l'icône de calcul).

5. Il s'agit des outils de base de tout éditeur graphique, je n'en explique pas le fonctionnement car ils font partie des pré-requis du cours.

6. Ces outils graphiques sont assez rudimentaires. Ils ne permettent pas de faire des rotations, des dégradés de couleur, etc... Authorware ne se substitue donc pas à un bon logiciel de dessin. Les graphismes sophistiqués seront conçus dans un logiciel spécifique de dessin et copiés-collés dans Authorware. La spécificité d'Authorware est d'assembler des éléments (graphiques, images, sons, ...) dans un scénario interactif.

menus Authorware (file, edit, data, attributes...) apparaissent en haut de l'écran, vous êtes en mode auteur. Voici les différentes façons de passer d'un mode à l'autre.

- En mode auteur, sélectionnez '**restart'** dans le menu 'Control' pour passer en mode utilisateur.
- En mode utilisateur, tapez control-p (ou '**pause**' dans le menu 'control') pour interrompre le programme et passer en mode auteur. A ce stade, l'option 'show current icon' affiche en contraste inversé l'icône qu'Authorware était en train d'exécuter au moment de l'interruption. Ensuite, vous sélectionnez l'option '**Play**' pour reprendre l'exécution (mode utilisateur). Vous pouvez aussi utliser la barre de contrôle qui reste affichée à l'écran si vous sélectionnez l'option 'control panel' dans le menu 'window'.
- En mode utilisateur, si vous cliquez deux fois sur un objet de l'écran, Authorware passe en mode auteur et édite l'icône à laquelle appartient l'objet sur lequel vous avez cliqué.
- En mode utilisateur, si Authorware rencontre ce qu'il considère comme un accident, par exemple une icône de présentation vide ou une icône dont un paramètre important n'a pas été déterminé (par exemple, une icône d'effacement pour laquelle vous n'avez pas déterminé ce qui devait être effacé), il passe automatiquement en mode auteur et édite l'icône qui pose problème. Attention, cette fonctionnalité est très pratique mais déroute les débutants qui se retrouvent inopinément en mode auteur sans l'avoir explicitement demandé.
- Enfin, vous pouvez demander à Authorware d'exécuter seulement une portion du programme. Sélectionnez le drapeau blanc, disponible dans la palette sous les icônes, glissez-le sur la barre de déroulement. Si vous choisissez l'option 'run from flag' dans le menu 'Control', Authorware exécutera uniquement le programme à partir de ce point. Si vous placez également le drapeau noir, Authorware interrompra l'exécution lorsqu'il arrivera sur ce drapeau. Le 'run from flag' est très utile pour tester des parties de programme sans devoir exécuter le tout.

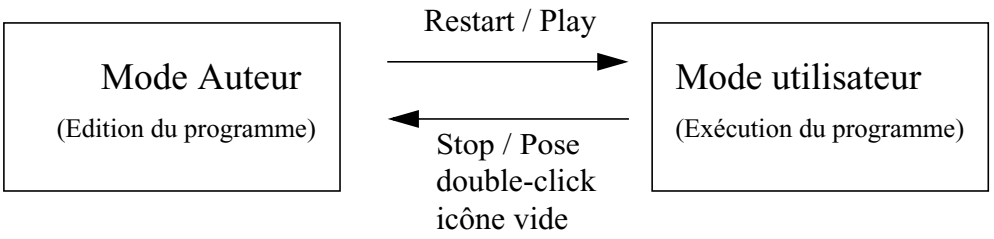


Figure 1-1:Comment passer du mode auteur au mode utilisateur et vice-versa ?

1-4 Mode vectoriel et mode 'calque'

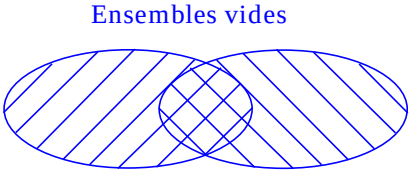
Authorware fonctionne en mode vectoriel pour l'auteur : les différents objets affichés sur un écran peuvent être modifiés individuellement par l'auteur lorsqu'il édite cet écran. Par contre, Authorware fonctionne en mode 'calque' lors de l'exécution du programme: les opérations programmées (effacement, animation, sélection d'objets) portent sur l'ensemble des objets d'une même icône de présentation. Si vous placez successivement plusieurs icônes de présentation, le contenu de ces icônes se superposera à l'écran comme si on y plaçait successivement différents calques transparents. Nous verrons plus tard comment ceci influence les animations et certaines modalités d'interaction. Vous pouvez néanmoins déjà en apprécier l'impact sur les mécanismes d'effacement: Authorware ne peut effacer partiellement le contenu d'une icône de présentation. Il efface tout ou rien. Aussi, si vous désirez effacer partiellement des informations, il faut placer celles-ci dans des icônes de présentation différentes, c'est-à-dire regrouper sur une même icône les objets qui seront effacés simultanément.

Les effets de la superposition des calques peuvent être influencés par le **mode de superposition** des objets. Pour changer les modes, choisissez **l'inspector** 'modes' dans le menu 'Window'. En général, on travaille avec des objets **opaques**: quand un objet est affiché, il masque les objets qui se trouvent préalablement au même endroit. Vous pouvez cependant modifier le mode grâce à l'option 'modes' du menu 'Window/Inspectos'. En mode '**transparent**', les pixels blancs de l'objet affiché prendront la couleur des objets auxquels il

se superpose (c'est-à-dire des objets affichés dans une icône placée plus haut dans le programme). Cette option est souvent utilisée pour afficher un texte sur un fond de couleur. Les autres modes, 'inversion' , 'erase' et ‘alpha’ (mélange de couleurs par transparence) sont plus rarement utilisés[7].

Exercice 1-2: Hachures

Construisez un programme qui présente un écran entièrement jaune, sur lequel apparaît un diagramme comprenant deux ensembles hachurés de telle sorte que ces hachures forment des croisillons à l'intersection des deux ensembles (Cf. ci-dessous). Au-dessus du diagramme, affichez le texte "Ensembles vides" en rouge.



Pour créer le fond jaune, vous avez sans doute placé un rectangle jaune aux dimensions de la fenêtre-utilisateur. En éditant l’icône de présentation, vous avez pu constater combien il est ensuite difficile de sélectionner les objets fins. En effet, vous avez certainement sélectionné involontairement le rectangle jaune pourtant en arrière-plan. Il existe une façon plus confortable de déterminer la couleur de fond de la fenêtre-utilisateur: sélectionnez l’item 'file properties' dans le menu 'modify', choisissez l’onglet ‘Playback’. Le champs **'colors'** vous permet de choisir une autre couleur pour l’arrière-plan. Cette option ne vous permet par contre pas de changer de couleur de fond en cours d'exécution du programme.

Enfin, nous noterez que ce même menu vous permet de modifier la taille de la fenêtre-utilisateur. Il est parfois pénible de passer du mode 'auteur' au mode 'utilisateur' parce que la fenêtre-utilisateur occupe tout l'écran et cache les icônes de l'auteur. Tant que vous vous familiarisez avec Authorware, je vous conseille de réduire la taille de la fenêtre utilisateur afin qu'elle ne recouvre pas entièrement la fenêtre avec vos icônes[8]. Pour modifier la taille de la fenêtre utilisateur, sélectionnez l’item 'file properties' dans le menu 'modify', choisissez l’onglet ‘Playback’'. Ensuite, l’item ‘size’ vour propose différents formats, choisissez le format **'variable'** qui est le moins contraignant.

1-5 Paramètres de l’icône d’effacement.

L’édition d’une icône d’effacement vous permet de régler plusieurs paramètres:

Vous pouvez effacer le contenu de plusieurs icônes en une fois, simplement en cliquant successivement sur des objets appartenant à des icônes différentes. Pour cela, il faut que tout ce que vous voulez faire effacer soit affiché à l’écran. Vous pouvez obtenir cet affichage de deux manières: (1) vous placez une icône d’effacement vierge (non éditée) dans le programme et vous exécutez celui-ci, Authorware s’arrêtera automatiquement à cette icône vierge pour vous demander ce qu’il faut effacer et aura auparavant affiché le contenu des différentes icônes de présentation; (2) vous double-cliquez sur la première icône dont le contenu doit être affiché, ensuite vous double-cliquez sur les autres en maintenant la touche ‘majuscules’ enfoncée.

---

7. En mode 'inversion', l'objet apparaît normalement s'il se superpose à un fond blanc, par contre, si le fond est coloré, les pixels colorés de l'objet sont inversés et les pixels blancs prennent la couleur du fond. En mode 'erase', l'objet est invisible si le fond est blanc; si le fond est coloré, l'avant-plan de l'objet est blanc et l'arrière plan est transparent.

8. Ceci est propre au fait que vous apprenez Authorware. Par contre, lors d'un développement, il est essentiel de déterminer une fois pour toutes le format de cette fenêtre car toute modification en cours de développement implique de réagencer tous les objets graphiques préalablement positionnés.

Exercice 1-3: Drapeau français

Réalisez un programme qui présente le drapeau français, puis fait une pause. Lorsque l'utilisateur presse 'continue', le système efface la bande rouge, puis fait une deuxième pause. Lorsque l'utilisateur presse 'continue', le système efface la bande blanche, puis fait une troisième pause. Lorsque l'utilisateur presse 'continue', le système efface la bande bleue.

Si vous désirez effacer le contenu de toutes les icônes affichées, ou presque toutes, vous pouvez procéder inversement: sélectionnez l'option **'Erase all icons except:'** et sélectionnez seulement les objets des icônes dont vous ne désirez pas effacer le contenu.

L'option 'effect' vous permet de choisir entre un effacement immédiat complet ('effect: none') et différentes formes d'effacement progressif. Vous pouvez tester différents effets d'effacement sur le même objet, en sélectionnant un nouvel effet et en cliquant sur le bouton 'replay'[9].

Notez au passage que les mêmes effets peuvent être utilisés pour afficher le texte (menu 'modify', item' icon transitions').

Exercice 1-4: Les quatre formes

Construisez un programme qui présente quatre formes géométriques de différentes couleurs. Le système affiche ces formes, puis fait une pause. Lorsque l'utilisateur presse 'continue', le système efface les quatre formes au moyen d'effets d'effacement différents

1-6 Paramètres de l'icône de temporisation (icône "WAIT")

Cette icône est très importante pour contrôler le déroulement du programme, en particulier pour laisser à l'utilisateur le temps de lire l'information présentée. Lorsque vous éditez cette icône, vous disposez de plusieurs façons de spécifier ce temps:

- Fixer un temps maximum (" Time limit "), en secondes. Il peut s'agir d'une fraction de seconde (par exemple, 0.1 s.). L'option 'show time remaining' affiche un symbole graphique qui informe l'utilisateur du temps restant[10].

Exercice 1-5: Patience

Construisez un programme qui affiche le message " Patience je calcule " pendant 5 secondes et informe l'utilisateur du temps écoulé .

- Attendre un signal de l'utilisateur, trois signaux sont disponibles :(1) l'utilisateur clique sur un point quelconque de l'écran ('Wait for mouse click '), (2) l'utilisateur presse une touche quelconque au clavier ('Wait for keypress') ou (3) l'utilisateur presse le bouton 'Continue'[11] (option 'show button').[12]
- Vous pouvez combiner les deux critères de temporisation, par exemple attendre que l'utilisateur presse le bouton 'continue' et fixer une limite de temps de 25 secondes. Dans ce cas, la pause sera

9. A propos de Design : L'utilisation fréquente d'effets spectaculaires d'effacement ou d'affichage peut s'avérer lassante voire perturbante
10. A propos de Design : Fixer une limite de temps absolue peut convenir lorsque l'information à présenter est brève, mais en général il est préférable de s'assurer que l'utilisateur a eu le temps de prendre connaissance de l'information présentée en le laissant presser le bouton 'Continue' lorsqu'il a fini de prendre connaissance de l'information présentée.
11. Le texte de ce bouton peut être modifié dans le menu 'modify', option 'file properties', champs 'label'. La position de ce bouton est par défaut identique au dernier bouton 'continue' que vous positionnez, afin de faciliter l'homogénéisation des présentations, mais il peut être modifié par l'auteur en cours d'exécution du programme, simplement en cliquant sur ce bouton et en le déplaçant (drag & drop)
12. A propos de Design : Si vous attendez que l'utilisateur fasse quelque chose, ce dernier doit en être averti, sinon le programme risque d'attendre longtemps. Si vous choisissez l'option 'show button', l'utilisateur voit effectivement le bouton 'continue' qui l'invite à presser ce bouton (selon une convention maintenant connue de la plupart des utilisateurs). Dans les autres cas, il est indispensable de mettre un message explicite tel que 'pressez une touche pour continuer'.

finie **dès que l'un des critères est satisfait**: si l'utilisateur presse 'continue', il reprend dès que le bouton a été pressé ; sinon, il reprend après 25 secondes.

Exercice 1-6: Temporisateur

Construisez un programme qui affiche deux pages de texte. Chaque page comprend 5 lignes. L'utilisateur presse le bouton 'continue' pour passer à la page suivante. S'il n'a pas pressé le bouton après une minute, passer automatiquement à la deuxième page.

Pour effacer ou animer (voir leçon 7) des éléments d'un même écran, indépendamment les uns des autres, ils doivent figurer sur des icônes différentes. Ceci pose des problèmes pour les positionner correctement les uns par rapport aux autres. Authorware offre quatre solutions à ce problème:

- Exécutez le programme de telle sorte à ce que tous les éléments apparaissent, puis double-cliquez sur celui que vous désirez positionner autrement.
- Utilisez la grille d'écran (menu 'view', option '**Grid**'). Celle-ci affiche à l'écran des repères qui vous permettent de positionner vos objets, mais qui ne seront pas visibles par l'utilisateur. Si vous sélectionnez en outre l'option 'Snap to grid', du même menu, tout objet placé dansl'icône sera déplacé vers la position de la grille la plus proche. Par contre, l'option '**Align'** (menu 'Modify') ne permet que d'aligner les objets d'une même icône.
- Si la grille que propose Authorware ne vous convient pas, créez votre propre grille: dessinez-la sur une icône de présentation que vous mettez en début de programme, qui vous servira pendant la phase de conception et vous supprimerez quand le programme sera au point.
- Tous les objets peuvent être positionnés au moyen de valeurs numériques (coordonnées horizontales et verticales), ce qui évite les imperfections d'un positionnement à la main. Ce point est expliqué dans la section "Positionnement variable sans animation", page 58.

1-7 Organiser son programme

Lorsque la barre de déroulement est complètement remplie d'icônes et que la fenêtre de votre programme atteint les limites de votre écran, il n'est pas possible de faire scroller l'écran vers le bas afin d'ajouter des nouvelles icônes. Aussi, il faut regrouper les icônes déjà positionnées en sous-programmes. Pour cela sélectionnez plusieurs icônes en les encerclant avec la souris (comme pour sélectionner plusieurs objets dans un logiciel de dessin) et choisissez l'option '**Group'** dans le menu 'Modify'. Authorware place alors une icône sous-programme à la place de celles que vous avez sélectionnées. Si vous double-cliquez sur ce sous-programme, Authorware affiche une fenêtre semblable à celle de votre programme principal et contenant les icônes que vous aviez encerclées. Vous venez de créer un sous-programme. Vous pouvez par la suite 'dégrouper' les icônes d'un sous-programme, c'est-à-dire les replacer dans le programme de niveau supérieur. Vous pouvez en outre directement créer un sous-programme en prenant un icône sous-programme dans la palette d'icônes (la dixième depuis le haut). Un sous-programme peut à son tour contenir des sous-programmes et ainsi de suite.

La présence de sous-programmes ne modifie pas l'exécution du logiciel. Si deux sous-programmes se suivent, la première icône du second sous-programme est exécutée juste après la dernière icône du premier sous-programme, comme si elles se suivaient sur une même barre de déroulement. L'utilisation de sous-programmes est surtout une forme d'hygiène de programmation: en matière de programmation graphique, il est important d'organiser ses icônes au sein d'une hiérarchie claire de sous-programmes. Il est aussi vivement recommandé de regrouper ses icônes selon des critères significatifs et de donner aux sous-programmes les noms les plus explicites possibles.

Leçon 2. Construire une interaction simple

2-1 Objectif

Dans cette leçon, vous apprendrez à créer une interaction simple dans laquelle l'utilisateur répond en cliquant sur un des boutons qui lui sont proposés.

2-2 Modalités d'interaction

Authorware dispose de 6 modalités d'interaction. L'utilisateur peut fournir une réponse:

1. en pressant un 'bouton' (type 'button');
2. en cliquant dans une zone sensible (type 'hot spot');
3. en cliquant sur un objet sensible (type 'hot object');
4. en déplaçant un objet (type 'target area');
5. en introduisant plusieurs caractères au clavier (type 'text entry');
6. en pressant une touche (type 'keypress');

Nous commençons par la modalité la plus simple à programmer, les réponses de type 'bouton' et verrons les autres par la suite. La plupart des paramètres à éditer sont identiques pour les différentes modalités.

2-3 Structure d'une interaction simple

Une interaction se compose généralement de trois étapes:

- poser une question, inviter l'utilisateur à faire un choix, ... bref solliciter implicitement ou explicitement l'activité de l'utilisateur ;
- permettre à l'utilisateur de réaliser une activité (cliquer, entrer des caractères au clavier, ...) et analyser cette activité ;
- selon l'analyse des actions de l'utilisateur, choisir une réaction appropriée et, selon les cas, solliciter un nouvelle fois la même activité ou passer à une autre.

La figure 2.1. illustre cette structure d'interaction. L'analyse de la réponse comporte une séquence de tests, représentés par des losanges. Chaque test permet d'identifier une catégorie de réponses. Dans une analyse simple, les tests sont disposés en cascade: les réponses qui appartiennent à la première catégorie donnent lieu au premier feed-back, seules celles qui n'appartiennent pas à la première catégorie sont ensuite testées par rapport à la deuxième catégorie et ainsi de suite. Selon la classe de réponses, l'utilisateur est invité à recommencer (la flèche remonte au point initial) ou passe à la suite (flèche sortant vers le bas).

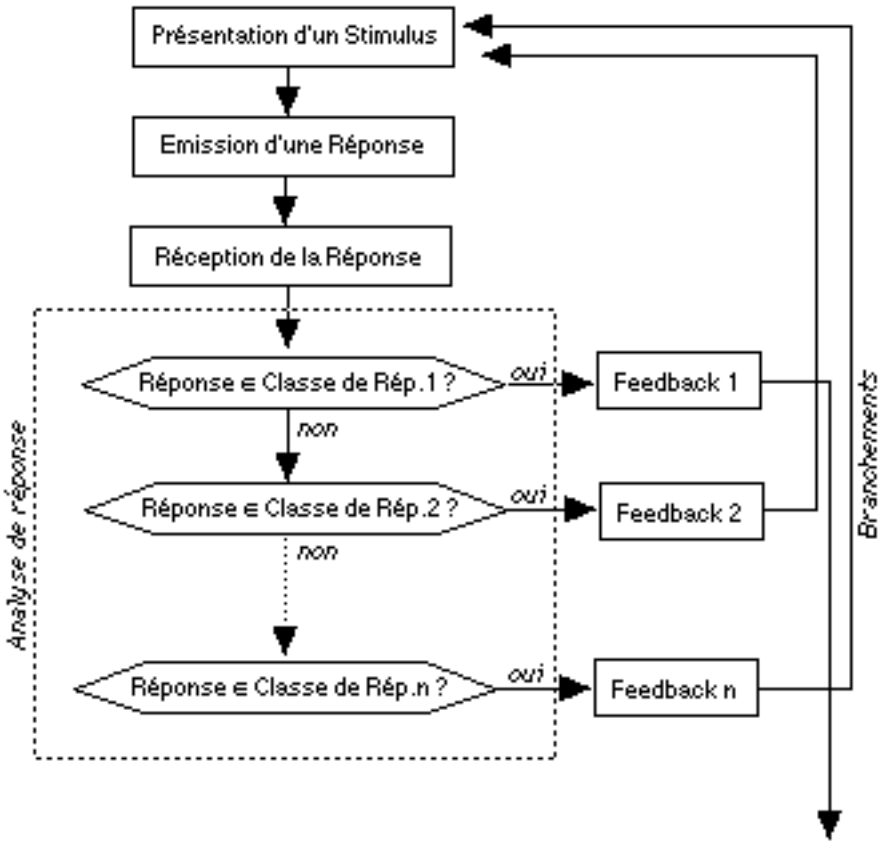


Figure 2-1:Structure d'une interaction

La représentation des interactions dans Authorware est très proche de l'ordinogramme présenté dans la figure 2.1. Si on applique à la figure 2.1 une rotation de 90 dans le sens des aiguilles d'une montre, on obtient la figure 2.2, qui correspond d'assez près à la représentation des interactions dans Authorware.

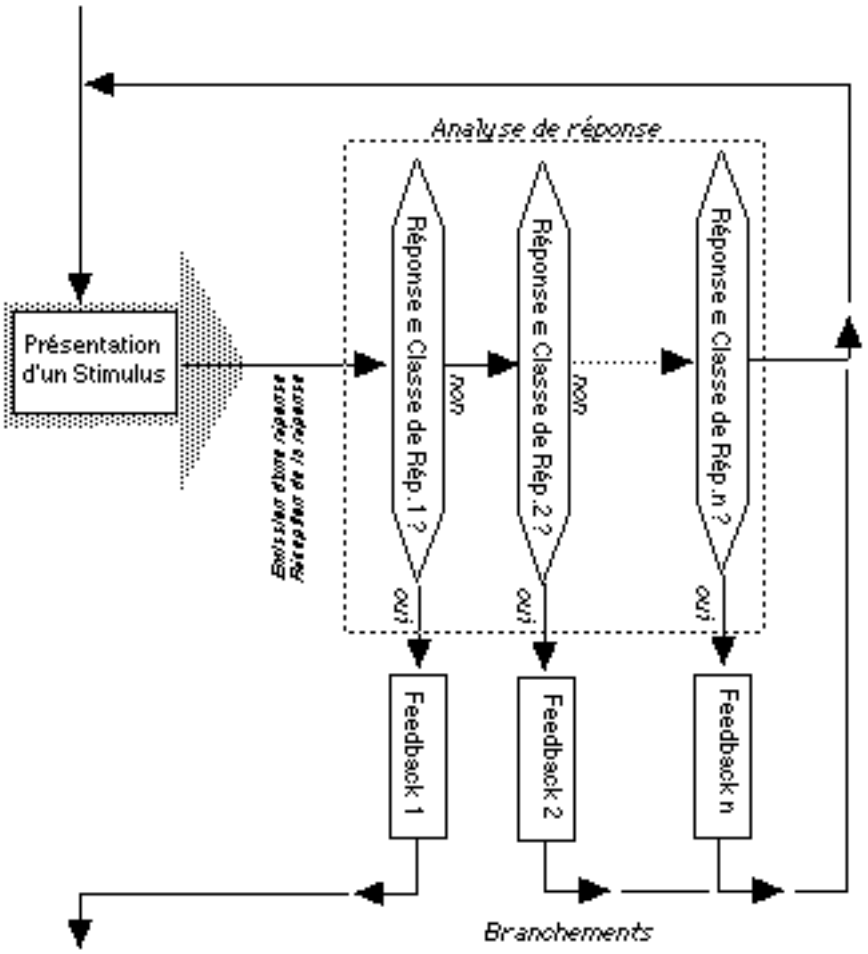


Figure 2-2:Transformation de l'algorithme 2.1 'à la Authorware'

Exercice 2-1: Une interaction simple

Construisons ensemble la question suivante " Quelle est la capitale des Pays-Bas? ". Nous allons traiter trois réponses, " Amsterdam "; " Rotterdam " et " La Haie ".

1. *Sélectionnez l'icône d'interaction (celle qui porte un ?) dans la palette d'icône et déposez-la sur la barre de déroulement.*
2. *Double-cliquez sur cette icône pour en éditer le contenu. Authorware ouvre une fenêtre dans laquelle vous allez présenter la question. Vous disposez des mêmes outils graphiques que pour l'icône de présentation.*
3. *Nous allons maintenant construire une par une chaque réponse et son feed-back. Prenez une icône de présentation dans la palette et posez là juste à droite de l'icône d'interaction, non pas sur la barre de déroulement mais sur la partie vierge du programme. Authorware dessine un petit algorithme et ouvre une boîte de dialogue dans laquelle il vous propose de choisir entre 11 modalités de réponses. Choisissez la première: 'button'. Si Authorware ne dessine pas ce petit algorithme, mais pose simplement l'icône sur la barre de déroulement, c'est que vous avez mal visé! Prenez l'icône et déposez-la à nouveau à droite de l'icône 'interaction'.*
4. *Cliquez deux fois sur le petit bouton qui figure dans l'algorithme juste au-dessus de l'icô-*

ne que vous avez déposée. Authorware ouvre une boîte de dialogue qui comporte de nombreux paramètres, répartis sur deux onglets. Les paramètres présents dans l'onglet 'response' seront communs à toutes les réponses. Les paramètres dans l'onglet 'button' sont spécifiques à ce type de réponse. Changez la valeur de 'label' (Untitled) (onglet 'button') en y tapant le nom de la réponse 'Amsterdam'.

5. *Ensuite double-cliquez sous l'icône de présentation placée sous ce bouton afin de taper le message que vous afficherez si l'utilisateur presse sur le bouton 'Amsterdam'. Affichez le feed-back " Non, Amsterdam n'est pas la capitale des Pays-Bas. Choisis une autre réponse "*
6. *Prenez une nouvelle icône de présentation et placez-la à droite du premier feed-back[13]. Donnez à ce deuxième bouton le nom 'Rotterdam', ensuite éditez le feed-back dans l'icône de présentation: " Non, Rotterdam n'est pas la capitale des Pays-Bas. Choisis une autre réponse ".*
7. *Recommencez (6) en créant un bouton " La Haie " en y associant le feed-back "En effet, c'est La Haie; on dit Den Hague en néerlandais "*
8. *Exécutez le programme. Si la position ou les dimensions des boutons ou des feed-back ne vous satisfont pas, vous pouvez les modifier en double-cliquant sur ces boutons. Choisissez ensuite l'option 'play' pour reprendre l'exécution.*
9. *Vous constatez que l'utilisateur recommence toujours la question quelle que soit sa réponse. Il conviendrait plutôt qu'il sorte de l'interaction en cas de réponse correcte. Aussi, double-cliquez sur le bouton " La Haie ". Dans l'onglet 'response'. dans le champs 'branch', sélectionnez l'option 'try again' qui détermine le fait que l'utilisateur recommence l'interaction. Cliquez sur cette option pour la modifier: choisissez l'option 'exit interaction'. Authorware modifie le dessin de l'algorithme en fonction de ce nouveau paramètre.*
10. *Exécutez le programme. Vous constatez que l'utilisateur n'a pas le temps de lire le feed-back correspondant à la bonne réponse. Pour introduire une temporisation à la sortie d'une interaction, il faut double-cliquer sur l'icône d'interaction et sélectionner les options 'Pause before exiting' et 'Show button'.*

Une caractéristique intéressante d'Authorware est que **la structure de l'interaction est indépendante du type de réponse**. Cette abstraction est liée au fait que les formes de branchement proposées sont identiques quelle que soit la modalité de réponse. Or ce sont ces formes de branchement qui déterminent la structure de l'algorithme. Dans l'exercice 2.1, vous avez découvert deux formes de branchement: " try again " pour recommencer l'interaction et " Exit interaction " pour reprendre l'exécution du programme sous l'icône d'interaction. Authorware vous propose deux autres modes de branchement " Return " et " Continue " que nous étudierons plus tard, mais faites bien attention à cette **erreur fréquente**: " Continue " ne signifie ni passer à la suite du programme (pour cela, sélectionnez l'option "exit interaction "), ni recommencer la question (pour cela, sélectionnez l'option " try again ").

13. Cette fois Authorware ne vous demande pas de préciser le mode de réponse car, par défaut, il construit un bouton aux paramètres semblables au précédent. Vous pouvez néanmoins modifier tous ces paramètres ainsi que le mode de réponse par la suite.

2-4 Gestion d'écran

L'effacement de la question, de la réponse ou du feed-back peut être réalisé manuellement au moyen de l'icône d'effacement. Toutefois, certaines icônes, dont l'icône d'interaction, offrent aussi une gestion automatisée de l'effacement au sein d'une interaction[14]. Nous vous recommandons d'utiliser le plus souvent possible cette gestion automatique des effacements.

2.4.1 Effacer la question

Lorsque vous éditez l'icône d'interaction, vous pouvez préciser le mode d'effacement (effets spéciaux) et le moment de l'effacement de la question (contenu de l'icône interaction):

- En fin d'interaction ('**upon exit**'): c'est la modalité la plus fréquente puisqu'il convient en général que l'utilisateur puisse relire la consigne ou la question à chaque tentative de réponse.
- Après chaque réponse du sujet ('**after next entry**'): cette option est par exemple utile lorsque la question occupe une zone de l'écran que l'auteur désire utiliser pour afficher le feed-back. La question sera réaffichée dès que l'utilisateur sera invité à répondre à nouveau. Dans ce cas il est préférable de placer une temporisation dans le feedback.
- Ne pas effacer ('**don't erase**'): la question sera effacée ultérieurement par une icône explicite d'effacement. Cette option est par exemple utile lorsque le texte de l'interaction est utilisé pour plusieurs questions consécutives.

2.4.2 Effacer le feed-back.

Lorsque vous double-cliquez sur le petit symbole représentant une réponse 'bouton' (ou de tout autre modalité), la boîte de dialogue vous permet de déterminer quand Authorware effacera le feed-back. Ce qui figure sur l'écran au terme du feed-back (quelle que soit sa durée) peut être effacé:

- Avant que le sujet fournisse une autre réponse ('**before next entry**'): cette option est utilisée lorsque le feed-back est un sous-programme comprenant d'autres interactions, des temporisations, etc. Par contre, si le feed-back est un simple message et que l'option 'try again' a été sélectionnée, le feed-back sera effacé immédiatement, l'utilisateur n'ayant le temps de lire ce feed-back;
- Après que le sujet fournisse une nouvelle réponse ('**after next entry**'): c'est l'option **la plus fréquente** car elle permet au sujet de lire le feed-back jusqu'au moment où il se décide à fournir une nouvelle réponse;
- Lorsque l'interaction est terminée ('**upon exit**'): cette option est très utile lorsqu'on désire par exemple que l'utilisateur puisse voir les différents feed-back qu'il a reçus au sein d'une même interaction;
- Ne pas effacer ('**don't erase**'): le feed-back sera effacé ultérieurement par une icône d'effacement. Cette option est utile lorsque l'auteur désire utiliser le feed-back fourni en dehors de l'interaction. Exemple: dans une première question le sujet choisit une photo d'oiseau, il sort de l'interaction mais la photo reste affichée pour la question suivante dans laquelle on lui demande le nom de l'oiseau.

14. A propos de Design : Effacer c'est retirer une information de l'écran. Cette lapalissade n'est pas gratuite: si cette information est utile, l'utilisateur ne disposera plus que de ce qu'il a spontanément gardé en mémoire. Les risques d'oubli sont proportionnels à la complexité de l'information effacée. Aussi, vous serez particulièrement attentif à **maintenir à l'écran l'information nécessaire à l'activité de l'utilisateur :** (1) effacer la question à chaque réponse prive le sujet de pouvoir relire la question pour comprendre le feed-back; (2) effacer la réponse avant que le sujet n'ait lu le feed-back prive le sujet de pouvoir comparer sa réponse et le feed-back fourni et (3) effacer le feed-back avant que le sujet fournisse une nouvelle réponse ne permet pas au sujet de relire le feed-back pendant qu'il réfléchit à sa nouvelle réponse.

Il n'est pas toujours facile de savoir à quel moment Authorware effacera le feed-back. La figure figure 2-3 illustre les trois temps d'effacement selon les réglages liés au feed-back "B".

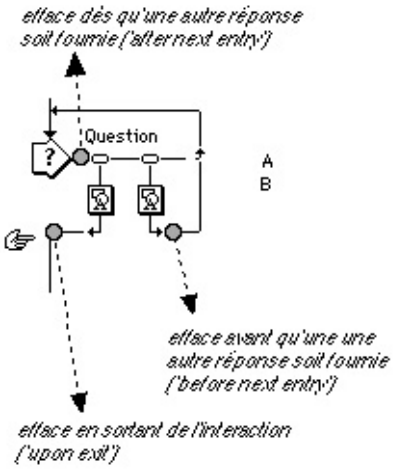


Figure 2-3:A quel moment se produit l'effacement?.

Exercice 2-2: Questions siamoises

Réaliser un programme qui comprend deux questions.

1. *Demander au sujet de cliquer sur les villes capitales. Afficher 8 boutons contenant les noms de villes suivantes: Zurich, Bern, New York, Washington, Ottawa, Montréal, Sidney et Camberra. Lorsqu'il clique sur un bouton, le feed-back consiste à afficher une marque verte à côté du bouton si c'est bien une capitale, sinon une marque rouge. Ne pas effacer ces marques de telle sorte que le sujet voie ses précédentes tentatives de réponses. Prévoir également un bouton 'j'ai fini' qui passe à la deuxième question. Ne pas effacer la première question dont les informations sont nécessaires par la suite.*
2. *Demander au sujet :" D'après les réponses ci-dessus, la ville du pays qui est la plus grande ou la plus connue est-elle toujours la capitale? ", prévoir deux boutons 'oui' et 'non' et fournir des feed-back adéquats.*

2-5 Compter les bonnes et mauvaises réponses

Le concepteur d'un logiciel pédagogique désire généralement compter le nombre de réponses correctes ou incorrectes de l'apprenant, soit pour adapter les interactions au niveau de l'apprenant, soit pour fournir à l'enseignant des informations sur le travail de l'apprenant. Aussi, Authorware compte lui-même les bonnes et mauvaises réponses, pour autant que l'auteur prenne soin de spécifier, pour chaque réponse, si elle est correcte, incorrecte ou neutre. Pour déterminer cela, double-cliquez sur le symbole représentant la réponse de type bouton et choisissez l'option '**correct response'**, '**wrong response'** or '**not judged'**. Selon votre choix, Authorware place un '+' ou un '-' devant le nom de l'icône correspondante dans l'algorithme d'interaction. Une réponse 'not judged' est une réponse que l'auteur ne désire compter ni dans la catégorie 'bonnes réponses' ni dans la catégorie 'mauvaises réponses'. Il s'agit par exemple, de réponses de type 'texte' que le système ne parvient pas à identifier et invite l'utilisateur à reformuler autrement . Il s'agit aussi des réponses aux questions non pédagogiques telles que 'quel est ton nom', 'veux-tu supprimer le son' , ...

Le nombre de réponses correctes et incorrectes est comptabilisé dans différentes variables que nous étudierons dans la leçon 6. ATTENTION: le choix de ces options ne modifie pas le déroulement de l'interaction. D'autre part, Authorware permet d'attribuer un score à chaque réponse, de fixer un score minimum pour considérer qu'on objectif est atteint, et un ensemble de fonctionnalités de gestion de l'apprentissage (voir par exemple le menu 'modify', option 'file properties', onglet CMI). Ces fonctions de **CMI**, propres à la version 5, ne sont pas décrites dans ce cours.

Vous devez toujours préciser pour chaque réponse si il convient de recommencer ou de sortir de l'interaction.

Exercice 2-3: Question négative[15]

Réaliser un programme qui pose la question " Clique sur l'affirmation qui est fausse ". Afficher deux boutons: " Tous les nombres pairs sont multiples de quatre " et " Tous les multiples de quatre sont des nombres pairs ". Fournir à l'utilisateur un feed-back approprié " Cette affirmation est effectivement fausse car 6 est pair mais n'est pas multiple de 4 " et " Non, c'est affirmation est correcte, car tout multiple de 4 est multiple de 4 est automatiquement multiple de 2 ". Préciser à Authorware quelle réponse est correcte et laquelle est incorrecte.

Comme vous l'observez dans l'exemple ci-dessus, la réponse correcte n'est pas toujours une proposition vraie, cela dépend de la formulation de la question. Après avoir exécuté le programme, vous pouvez vérifier le comptage des bonnes et mauvaises réponses en choisissant l'option 'variables' dans le menu 'window'. Sélectionnez la catégorie 'interaction', puis sélectionnez la variable TotalCorrect ou TotalWrong et leur valeur ('Current Value') sera affichée sur la droite.

2-6 Autres options du mode de réponse 'bouton'

Lorsque vous éditez un bouton, sous l'onglet 'button', Authorware vous offre encore d'autres options:

- Dans le champ **'keys'**, l'auteur introduit un raccourci-clavier pour le bouton en cours d'édition: par exemple, si vous déclarez A comme 'optional key' du bouton 'AIDE', si l'utilisateur presse commande-A, le programme founira le feed-back associé au bouton AIDE[16].
- L'option '**Make Default**' définit un raccourci-clavier particulier: si l'utilisateur presse sur RETURN, Authorware considère qu'il a pressé sur le bouton défini comme bouton par défaut. Celui-ci apparaît à l'écran encerclé d'un trait plus épais.[17]. Il ne peut bien sûr y avoir qu'un seul bouton par défaut par interaction, sinon Authorware ne saurait pas quel bouton est associé au signal 'RETURN'. Malheureusement Authorware ne permet pas de changer en cours d'exécution le choix du bouton par défaut. Cette fonctionnalité serait utile pour le cas où l'utilisateur traverse rapidement une séquence de choix de paramètres pour lesquels il désire conserver la dernière valeur sélectionnée.
- Par défaut, Authorware propose un bouton classique. En cliquant sur le bouton '**buttons**', vous pouvez sélectionner tout autre type de bouton dans la librairie de boutons. Nous discutons les différents types de boutons dans la section 2-8.
- En général, l'auteur positionne le bouton et adapte ses dimensions au moyen de la souris (il suffit de double-cliquer sur ce bouton en mode 'utilisateur'). Sachez que vous pouvez aussi définir position et dimension en introduisant les paramètres numériques[18]: position du bord gauche, position du bord supérieur, hauteur et largeur du bouton. Cette option est utile pour aligner les boutons avec précision ou pour s'assurer qu'ils ont la même taille. En effet, cette précision est difficile à obtenir lorsque l'auteur manipule le bouton avec la souris[19].

15. A propos de Design : Il est généralement recommandé de ne pas utiliser de formulation négative dans les questions et a fortiori de double négations. En effet, en cas d'erreur dans cet exemple on ignore si le sujet ne maîtrise pas la formulation de la question. En docimologie, on dirait que de telles questions ont une faible validité.

16. A propos de Design : Ces raccourcis-clavier sont intéressants si l'utilisateur est confronté à un même jeu de boutons pendant une durée assez longue pour qu'il puisse progressivement mémoriser le raccourci-clavier correspondant à chaque bouton. Par contre, un raccourci-clavier est inutile pour une interaction dans laquelle l'utilisateur ne fournirait qu'une réponse, un seule fois.

17. A propos de Design : Le bouton par défaut est utile lorsqu'on pose régulièrement une même question à l'utilisateur, par exemple, la confirmation d'une commande, avec des paramètres optionnels. Son utilisation requiert davantage de précautions dans une question pédagogique puisqu'elle pousse l'utilisateur vers une réponse particulière.

18. Au lieu de fournir des valeurs numériques précises, vous pouvez introduire des variables, de telle sorte que la position ou la taille du bouton soient déterminé dynamiquement en cours d'exécution. L'utilisation des variables sera abordée à partir de la leçon 6.

19. A propos de Design : Il est par exemple important que différents boutons ayant la même fonctionnalité soient positionnés au même endroit même s'ils appartiennent à différentes interactions . L'utilisation des coordonnées numériques du bouton permet de s'assurer facilement que deux boutons apparaîtront au même endroit.

- L'option 'scope: **perpetual**' (onglet 'response') existe pour la plupart des modalités de réponse. Elle permet dans ce cas de créer un bouton qui n'est pas simplement actif à un moment particulier du logiciel, mais reste disponible tout au long du logiciel. Nous étudierons les interactions perpétuelles dans la leçon 10, car celles-ci reposent sur des structures d'interaction différentes de celles que nous étudions pour le moment.
- Enfin, l'option '**active if true**' permet de rendre un bouton non-disponible pendant certaines phases du logiciel. Pour décider si le bouton est disponible à un moment particulier, Authorware évalue l'expression que l'auteur place dans le champ 'active if true', et désactive le bouton si cette expression est fausse et le réactive si elle est correcte. Une expression consiste généralement à comparer la valeur d'une variable et d'une constante ou de plusieurs variables, aussi nous reviendrons sur ce point après la leçon sur les variables. L'auteur peut en outre spécifier si un bouton inactif doit être invisible ou simplement apparaître en gris (option 'Hide when inactive')[20].

Exercice 2-4: Boutons par défaut

Poser la question " Voulez-vous vraiment effacer ce document ? " et afficher deux boutons: 'Annuler' et 'Ok', respectivement associés au feed-back 'document non effacé' et 'document effacé'. Le bouton par défaut est 'annuler'.

2-7 Biais pédagogique dans la terminologie

Dans ce cours, nous décrivons souvent les interactions en termes de question, réponse et feed-back, car Authorware constitue avant tout un outil de construction de didacticiels. La section précédente exprime bien ce biais, avec la notion de réponse correcte ou incorrecte, un concept qui n'est pas pertinent dans des applications non-pédagogiques, telles qu'un logiciel de réservation de billets de cinéma. Aussi, dans la suite du cours, les termes 'questions', 'réponse' et 'feed-back' doivent être entendus de manière très large:

- Une question peut être toute sollicitation de l'activité du sujet, par exemple " Quel jeu voulez-vous jouer ? ". Elle peut être implicite: par exemple, si l'activité de l'utilisateur consiste à attraper des objets qui se déplacent à l'écran, la sollicitation se réduit au déplacement de l'objet. Autre exemple, Authorware conçoit un menu comme une question implicitement posée en permanence (nous parlerons d'interaction perpétuelle), pourtant la présence du titre du menu dans la barre des menus s'éloigne de ce qu'on entend généralement par 'question'.
- Une réponse de l'utilisateur n'est pas nécessairement une réponse au sens pédagogique comme dans l'exemple géographique ci-dessus. Il peut s'agit par exemple de presser les touches de direction dans un jeu de type 'Tetris' ou de cliquer sur un item dans un menu.
- Un feed-back n'est pas nécessairement une évaluation pédagogique de la réponse de l'utilisateur, mais toute réaction du système spécifique à l'activité de l'utilisateur. Par exemple, 'déplacer un bloc vers la droite' est le feed-back que Tetris fournit si l'utilisateur presse sur la touche 'flèche vers la droite'. Le feed-back n'est pas nécessairement un message tel que dans l'exercice 2.1, il peut s'agir d'un son, d'une autre question, voire de tout un sous-programme. Par exemple, si la question est " Quelle leçon désirez-vous suivre " , le feed-back est constitué de la leçon sélectionnée.

Exercice 2-5: Questions emboîtées

Afficher le verbe 'voler'. Poser la question "A quel mode faut-il le conjuguer " et offrir deux réponses. Si l'utilisateur choisit le bouton 'indicatif', poser la question " A quel temps?" et offrir trois boutons 'présent', 'imparfait' et 'future simple'. Si l'utilisateur choisit le mode conditionnel, poser la question " A quel temps?" et offrir deux boutons 'présent' et 'imparfait'. Pour chaque choix afficher le verbe au mode et au temps choisi. Après avoir affiché le verbe conjugué, revenir à la question de départ.

20. A propos de Design : La différence entre un bouton gris ou invisible n'est pas anecdotique. Lorsque le bouton est affiché en gris, l'utilisateur perçoit que l'interaction conserve sa structure, au sein de laquelle certains items sont simplement désactivés. Par contre, si le bouton est invisible, il y a moins de chances que l'utilisateur perçoive ces invariants de l'interaction .

2-8 Types de bouton et librairie de boutons

Jusqu'à présent, vous avez utilisé un type de bouton standard qui comporte deux états, dans la terminologie Authorware, l'état 'up' et l'état 'down', 'down' étant la représentation graphique au moment où l'utilisateur clique sur le bouton. Pour un bouton standard, le graphisme de l'état 'down' est simplement l'inversion du graphisme de l'état 'up'.

Après avoir été sélectionné, le bouton standard retourne à son état initial, appelé état 'normal' dans la terminologie Authorware. D'autres boutons par contre changent d'apparence après que l'utilisateur les ait sélectionnés. C'est le cas des cases à cocher ('check box') et des boutons-radio ('radio button') : une croix apparaît dans une case qui a été cochée, alors que le bouton-radio se noircit. Dans la terminologie Authorware, le bouton est alors dans l'état 'checked'. Si l'utilisateur sélectionne à nouveau un bouton dans l'état 'checked', il revient dans l'état 'normal'. Ces boutons[21] permettent à l'utilisateur de visualiser ses choix préalables, ils ont donc une fonction de mémorisation de l'état des interactions[22]. En général, ces boutons ne sont pas suivis d'un feed-back immédiat, mais utilisés pour fixer des paramètres qui influenceront les interactions ultérieures. Aussi, nous utiliserons ces boutons au cours d'exercices ultérieurs, lorsque vous aurez appris à stocker des informations (au moyen de variables) afin de les réutiliser par la suite.

Pour modifier le type de bouton, cliquez sur le carré 'button type' dans la boîte d'édition de cette réponse. Authorware vous propose de sélectionner un type de bouton au sein d'une librairie de boutons. Au départ, cette librairie ne comprend que les trois types de boutons décrits précédemment (boutons standard, 'check box' et bouton-radio). Vous pouvez néanmoins créer d'autres types de boutons, comme nous l'expliquons dans la section suivante.

2-9 Comment créer de nouveaux boutons

En créant des boutons spécifiques, vous pouvez communiquer **implicitement** des informations à l'utilisateur quant aux modalités d'interaction ou aux effets du bouton. Pour créer un nouveau bouton, vous devez entrer dans la librairie de boutons, soit en cliquant sur 'button type' dans la boîte de dialogue permettant d'éditer une réponse de type 'bouton', soit en choisissant l'option '**buttons...**' dans le menu 'libraries'. Une fois dans la librairie, cliquez sur le bouton 'new'.

Créer un bouton consiste à définir les huit états graphiques possibles du bouton. A l'état 'normal', le bouton peut être 'up' (relâché), 'down' (enfoncé), 'over' (état du bouton quand le curseur passe sur le bouton) ou 'disabled' (inactif). Lorsque le bouton est à l'état 'checked', on retrouve les même quatre conditions, ce qui nous donne huit graphismes pour le même bouton. Ces différents graphismes sont en général différentes versions d'un même objet graphique: inversion des couleurs, déplacement d'un élément du graphisme... Ces graphiques peuvent être crées dans un logiciel graphique quelconque et ensuite être importés dans Authorware (option '**import**' dans la fenêtre permettant de créer un nouveau type de bouton). Chaque état du bouton peut en outre être associé à un son particulier, qui sera également créé hors Authorware et ensuite associé à un état du menu via l'option 'import'. Enfin, vous pouvez spécifier le texte éventuel associé au bouton ainsi que sa position par rapport au graphisme.

---

21. A propos de Design : Les boutons 'check box' et 'radio boutons' sont souvent utilisés selon des conventions différentes. Les 'check box' rappellent la métaphore de la 'check list' dans laquelle l'utilisateur peut 'marquer' plusieurs items. Par contre, les boutons-radio sont généralement utilisés pour faire des choix exclusifs (comme les différents canaux d'une radio): les boutons-radio sont regroupés en 'clusters' au sein desquels un seul bouton à la fois peut-être dans l'état 'checked'. Aussi la sélection d'un bouton radio implique automatiquement la dé-sélection au sein du même cluster du bouton qui était préalablement dans l'état 'checked'. ATTENTION, Authorware ne respecte pas ces conventions: la sélection d'un radio-bouton ne déselectionne pas les boutons préalablement sélectionnés. Si vous désirez respecter cette convention, vous devez prévoir dans chaque feed-back de modifier manuellement l'état des autres boutons au moyen d'une fonction: 'Checked@"iconTitle" := TRUE'. La signification et l'usage d'une telle commande est expliquée dans la leçon 6.

22. Il peut s'agir de mémoire externe pour un utilisateur unique ou de mémoire partagée, donc de moyen de coordination, lorsque plusieurs personnes utilisent simultanément le même logiciel

Leçon 3. La manipulation directe

3-1 Objectif

Dans cette leçon, vous apprendrez à construire des interactions dans lesquelles l'utilisateur répond en sélectionnant une zone de l'écran ou en déplaçant un objet.

3-2 Les micro-interactions

Vous avez déjà appris la modalité 'bouton' et apprendrez dans cette leçon les modalités 'zone sensible', 'objet sensible' et 'objet à déplacer'. La plupart des options de réponse sont identiques d'un type de réponse à un autre: mode de branchement, mode d'effacement du feed-back, possibilité de désactiver une réponse, réponse perpétuelle, ... Vous découvrirez ci-après d'autres paramètres qui varient selon le type de réponse. Ces paramètres définissent ce que nous appelons des micro-interactions, c'est-à-dire des événements qui interviennent entre la question et le feed-back.

Dans la leçon précédente, nous décrivions une interaction par un triplet: une sollicitation de la part du système, la réponse de l'utilisateur et les feed-back produits par le système. Néanmoins, en regardant de plus près, une interaction peut se décomposer en 9 étapes. Nous illustrons ces micro-interactions dans le cas des réponses de type 'bouton ' que vous connaissez déjà.

1. Présenter le **stimulus**. Le terme 'stimulus' recouvre non seulement les questions prises au sens pédagogique du terme, mais également tout événement qui déclenche une activité de l'utilisateur, par exemple l'apparition d'un ennemi dans un jeu ou la présence d'un menu dans la barre de menus.
2. Informer l'utilisateur quant à la **modalité de réponse**. Le sujet doit-il répondre en cliquant sur un objet, en déplaçant un objet, en entrant un texte au clavier,...? Dans le cas des interactions de type 'bouton', la simple présence des boutons dispense l'auteur de prévoir une consigne. Par contre, dans d'autres cas, il sera nécessaire d'ajouter un message explicite tel que "clique sur ...". Les réponses de type 'bouton' offrent deux variations de la modalité standard: (1) le bouton par défaut est activable pas la touche 'return', une convention que de nombreux utilisateur connaissent car elle est utilisée dans la plupart des logiciels et (2) le raccourci-clavier que l'utilisateur ne connaît a priori pas et dont il doit donc être informé.
3. Informer l'utilisateur quant aux **réponses possibles**. La présence de boutons indique intuitivement à l'utilisateur quelles sont les réponses possibles. Nous avons vu dans la leçon 2 que certains boutons ou items de menu peuvent être désactivés, soit en affichant ces boutons en gris, soit en ne les affichant simplement pas. Dans le premier cas, l'activation/désactivation est explicite; dans le second cas, elle est implicite. Lorsque le sujet dispose de plusieurs possibilités de réponse, il est en outre utile de préciser si ces réponses sont complémentaires ou exclusives.
4. L'utilisateur élabore sa réponse. Il doit notamment savoir quelles sont les modalités **d'édition de sa réponse**. Par exemple, dans le type de réponse 'bouton', si l'utilisateur clique sur un bouton mais déplace le curseur hors du bouton avant de relâcher le bouton de la souris, la sélection du bouton est annulée. Ces possibilités seront importantes pour les modes de réponses plus complexes.
5. L'utilisateur émet sa réponse : l'utilisateur construit sa réponse, la modifie éventuellement, jusqu'au moment où il réalise un acte particulier défini comme **signal d'émission**. Dans le cas des réponses de type 'bouton', le signal d'émission de la réponse est le fait de relâcher le bouton de la souris (et non de l'enfoncer comme on le croit souvent).
6. Le système reçoit la réponse introduite par l'utilisateur. Ce dernier est généralement informé de la bonne réception de sa réponse par la machine grâce à un **signal de réception**. Par exemple, dans l'interaction de type 'bouton', le bouton sélectionné s'inverse momentanément pour signifier à l'utilisateur que le système a bien perçu sa réponse et vas y donner suite prochainement[23].
7. Le système **analyse la réponse**. Le processus d'analyse de réponse est souvent réalisé par une séquence

de tests permettant d'identifier à quelle classe de réponses appartient la réponse de l'utilisateur. Une classe peut être définie en extension: "5 ou -5". Une classe peut bien sûr ne contenir qu'une seule réponse. Une classe peut aussi être définie en compréhension: par rapport à une réponse-type (la réponse 'hydrogène' et ses variations orthographiques raisonnables), au moyen des critères numériques (p.ex., l'ensemble des nombres entiers pour un distributeur de billets) ou de critères plus complexes (l'ensemble des dates plausibles dans un logiciel de réservation). La succession des tests qui permettent d'identifier la classe à laquelle appartient une réponse doit respecter une logique précise que nous décrivons en détail dans la leçon 4. Dans le cas des réponses de type bouton, l'analyse de réponse est simple, puisque l'utilisateur ne peut presser deux boutons simultanément.

8. Le système associe un **feed-back** à chaque classe de réponses. Le terme feed-back recouvre non seulement des messages de type "C'est juste", mais également toute autre activité sélectionnée par le système sur base de la classe de réponses identifiée: par exemple, poser une sous-question, choisir une chapitre, imprimer un document, effacer un fichier,... Cette étape est indépendante du type de réponse (bouton, objet sensible, texte, ..)
9. L'auteur détermine un **branchement** pour chaque classe de réponses. Le branchement détermine à quel point sera reprise l'exécution du programme après le feed-back. Dans un didacticiel, les branchements les plus fréquents consistent à reposer la question ou à passer à la question suivante. Cette étape est indépendante du type de réponses.

3-3 Les réponses de type 'zone sensible'

L'interaction se construit exactement comme les réponses de type bouton, mais au lieu d'afficher un ou plusieurs boutons rectangulaires à l'écran, l'auteur définit une ou plusieurs zones rectangulaires au sein duquel l'utilisateur devra cliquer. Lorsqu'Authorware vous invite à choisir le type de réponse, choisissez **'hot spot'**. Notez que si le système dispose d'un écran tactile, l'utilisateur pourra sélectionner les zones définies par simple pression du doigt sur l'écran, sans que vous ne deviez changer votre programme[24].

Examinons les micro-interactions spécifiques dans le cas des zones sensibles.

- Pour informer l'utilisateur quant aux réponses possibles, vous pouvez changer la **forme du curseur** lorsque celui-ci passe sur une zone sensible (option '**custom cursor**'). La modification du curseur au-dessus des zones de réponse prévues permet à l'utilisateur de détecter **les réponses possibles** à chaque changement de forme du curseur. Le curseur de substitution est choisi parmi une petite librairie de curseurs disponibles sur le système utilisé. Cette librairie peut être enrichie par le concepteur (Cf. "Librairie de curseurs", page 27)
- L'option '**mark after match**' affiche une marque noire à gauche de chaque zone que l'utilisateur sélectionne. Celui-ci est donc informé des zones qu'il a précédemment sélectionnées. Cette information est précieuse lorsque le sujet doit, pour une même question, fournir plusieurs réponses espacées dans le temps (par exemple, choisir les chapitres dans une table des matières). Cette information n'est malheureusement pas effaçable en cours d'interaction et n'est pas modifiable par l'auteur[25] (qui voudrait par exemple la placer ailleurs ou en changer la forme).

Exercice 3-1: Zones sensibles

Créez un programme qui présente quatre rectangles dont un carré. Demander à l'utilisateur de cliquer sur le carré. Utilisez le curseur 'main' pour la sélection des objets. En cas d'erreur, effacer le feed-back mais laisser une marque à côté des formes préalablement sélectionnées.

---

23. Dans un clavier normal, l'utilisateur sait qu'il a pressé la touche par un simple feed-back tactile. Dans les claviers à effleurement (photocopieuses, automates aux pompes à essence, ...), l'absence de signal tactile de réception de la réponse est généralement compensé par un signal sonore.
24. A propos de Design : Pour l'utilisation d'un écran tactile, il faut définir des zones sensible assez grandes, surtout sur l'axe vertical, car la désignation avec la main est moins précise que la désignation par la souris .
25. Si ces contraintes ne vous conviennent pas, il est relativement simple de les re-programmer. Il suffit d'ajouter dans chaque feed-back l'affichage d'un symbole quelconque, lequel n'informe pas de la qualité de la réponse, mais simplement de son choix.

- Vous pouvez modifier le **signal d'émission** de la réponse en choisissant le nombre de 'clicks' nécessaires. Lorsque vous éditez une zone sensible, l'option **'match with'** offre trois signaux: la réponse est émise (1) dès que le curseur passe sur la zone ou l'objet sensible (option **'cursor in area'**), (2) lorsque le sujet clique une fois (option **'single click'**) ou (3) lorsque le sujet clique deux fois (option **'double click'**). Ces différents signaux permettent au concepteur de discriminer différents niveaux d'intentionnalité: le sujet peut par exemple cliquer une fois pour obtenir de l'information sur un objet et deux fois pour désigner celui-ci comme réponse[26].
- Vous pouvez également spécifier le signal de réception de la réponse: sélectionnez l'option 'auto highlight' et lorsque le sujet sélectionne une zone, celle-ci apparaîtra une fraction de seconde en contraste inversé[27].

Exercice 3-2: Niveaux d'intentionnalité

Modifier le programme 3.1. pour ajouter les fonctionnalités suivantes. Lorsque le sujet passe le curseur au-dessus d'une forme, les dimensions de celle-ci sont affichées le long de ses côtés. En effet, il est relativement difficile de percevoir une petite différence de longueur entre deux côtés, en particulier parce que la vitre qui protège de l'écran maintient un écart d'un centimètre entre la figure représentée et la règle. Effacer les dimensions dès que le sujet déplace le curseur vers une autre forme. Si le sujet clique sur la forme, conserver les feed-back de la version 3.1. Inverser la couleur des formes lorsque le sujet clique dessus, mais pas lorsqu'il passe simplement dessus avec le curseur.

- Tous les autres options de ce type de réponses sont identiques aux réponses de type 'bouton'. Par contre, l'analyse de réponse offre une nouveauté. En effet dans le cas des réponses 'bouton', le sujet ne peut forcément cliquer que sur un bouton à la fois. Par contre, **des zones sensibles peuvent se recouvrir**. Si le sujet clique sur un point qui appartient à deux zones Z1 et Z2, quelle réponse sera considérée par Authorware? La réponse est simple: Authorware examine ces zones dans l'ordre où vous les placez dans l'icône d'interaction, de la gauche vers la droite. A chaque zone, il effectue un test : le clique du sujet se trouve-t-il dans cette zone? Si oui, il exécute le feed-back; si non, il examine la zone suivante. Aussi, si Z1 est placé avant (=à gauche) de Z2 dans l'algorithme d'analyse de réponse, c'est la réponse Z1 qui sera identifiée. Nous étudierons ces problèmes d'ordre des réponses en détail dans la leçon 4.

Exercice 3-3: Zones imbriquées

Créez un programme qui présente une carte (schématique) de la Suisse et demander à l'utilisateur de cliquer sur Berne. Prévoir trois feed-back: un pour la réponse exacte, un pour une zone proche de la réponse exacte ("C'est bien dans cette région, sois plus précis") et un pour toute autre réponse (n'importe où sur la carte).

3-4 Gestion d'écran

Dans ce genre d'exercices, vous devez régulièrement déplacer vos feed-back pour éviter qu'ils couvrent une information importante ou soient couverts par un autre message. Cette difficulté sera d'autant plus importante par la suite que différents objets à paraître sur le même écran seront créés dans des icônes différentes. Je vous recommande par la suite de travailler avec une grille d'écran: dans le menu 'View', sélectionnez l'option **'grid'** pour afficher la grille et **'snap grid'** pour contraindre le curseur à se positionner uniquement sur les points de la grille. Cette grille, non-visible par l'utilisateur, vous permet de définir à l'avance des zones spécifiques pour les différentes informations[28].

---

26. Il est difficile de tester votre programme quand vous choisissez l'option 'double-click' puisqu'un double-clique est aussi ce qui permet de passer du mode utilisateur au mode auteur. Sachez toutefois que ce problème disparaîtra lorsque vous générerez une version runtime ('package'), dans laquelle il n'est évidemment pas possible de passer en mode auteur.

27. A propos de Design : Nous recommandons d'utiliser ce signal de réception qui confirme à l'utilisateur que sa réponse a bien été perçue par le système. Ce signal est d'ailleurs systématique pour les réponses de type 'bouton'.

3-5 Les réponses de type 'objet sensible'

Les réponses de type 'objet sensible' (**hot object**') sont très semblables du point de vue de l'utilisateur aux réponses de type 'zone sensible'. Une zone sensible n'est pas visible pour l'utilisateur. Elle permet de demander à l'utilisateur de cliquer dans une zone qui ne correspond pas à un objet précis, tels que les angles dans la figure 3.1.a. La figure 3.1.b. illustre une limitation du mode 'zone sensible': les zones étant rectangulaires, la zone incluant le segment B et celle qui inclut le segment C se superposent. Dans ce cas, si le sujet clique dans l'intersection des zones B et C, l'auteur ne peut déterminer si le sujet a répondu B ou C. Le type de réponse 'objet sensible' résout ce problème en permettant de définir les objets B et C comme réponses (la réponse est identifiée si le sujet clique sur un pixel de ces objets)[29].

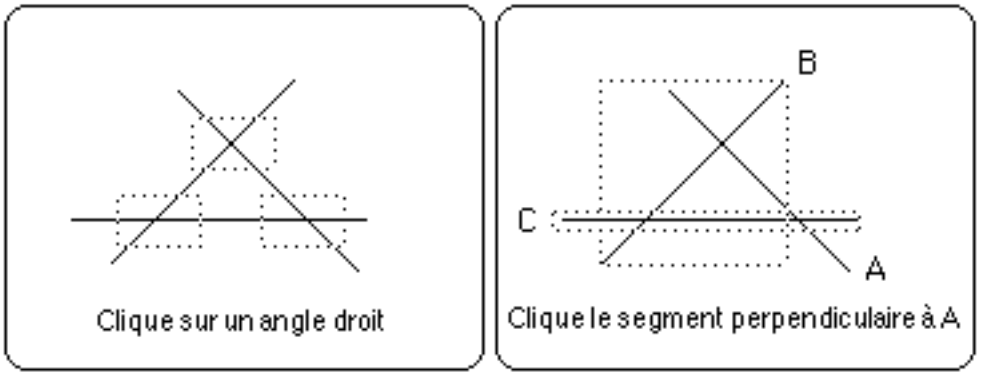


Figure 3-1:(a) A gauche: Utilisation pertinente du mode 'zone sensible' (b) A droite: Utilisation inadéquate du mode 'zone sensible' (Les zones de réponse représentées en pointillé ne sont en réalité pas visibles par l'utilisateur).

Attention, la création de réponses de type 'objet sensible' est différente de ce nous avons vu: **chaque objet sensible doit être créé dans une icône de présentation différente**. Il doit être seul dans cette icône. Les différentes icônes de présentation seront placées juste en amont de l'icône d'interaction. Lorsque vous éditez une réponse de type objet sensible, Authorware vous demande de sélectionner l'objet-cible: "target object: none (click an object)". Cliquez sur cet objet et le nom de l'icône dans laquelle il figure apparaîtra dans la boîte de dialogue permettant d'éditer cette réponse.

Exercice 3-4: La perpendiculaire

Créez un programme qui construit l'interaction décrite par la figure 3.1b. Le sujet doit répondre en cliquant sur B ou sur C.

Lorsque vous exécutez le programme, vous pouvez non seulement cliquer sur le segment B mais aussi sur la lettre B[30]. Rappelons qu'Authorware fonctionne en mode 'calque': la réponse 'segment B' ne comprend pas seulement le segment B mais tout pixel appartenant à l'icône dénommée 'segment B'.

D'autre part, vous avez aussi pu constater dans le programme 5.0 qu'il n'est pas facile de cliquer sur ces segments. Vous pouvez, comme pour les réponses de type sensible modifier le curseur afin de choisir un curseur adapté aux désignations précises (par exemple la croix). Néanmoins, la sélection d'un objet dont la largeur n'est que d'un pixel demande une excellente dextérité. Pour éviter les problèmes qui en résultent, il faut soit définir des objets plus larges, soit dessiner derrière l'objet visible un objet sensible plus large et invisible par l'utilisateur.

---

28. Cette grille facilite non seulement la programmation. Elle favorise aussi la création d'invariants dans la présentation des informations, un principe sur lequel nous reviendrons souvent dans ce cours .
29. Un autre avantage des objets sensibles est que ceux-ci peuvent être en mouvement, option notamment intéressante pour les jeux. Nous y reviendrons en étudiant l'animation (leçon 7).
30. Pour autant que cette lettre se trouve bien dans la même icône que le segment.

Exercice 3-5: La perpendiculaire saisissable

Modifiez le programme 3.5 et sauvez le sous 3.6. Ajoutez dans les deux icônes de présentation 'Segment B' et 'Segment C' un segment épais et invisible caché sous le segment sur lequel l'utilisateur doit cliquer.

En ce qui concerne le réglage des micro-interactions, les réponses de type 'objet sensible' disposent exactement des même paramètres que les zones sensibles, notamment les modifications du curseur, le nombre de clicks nécessaires et l'inversion momentanée de l'objet sélectionné.

3-6 Les réponses de type 'objet mobile'

Ce type de réponse, très utilisé dans la manipulation directe, consiste pour l'utilisateur à déplacer un objet dans une zone particulière (ou plusieurs objets dans plusieurs zones). Du point de vue de l'auteur, ce type de réponse est un peu plus compliqué à construire. Primo, vous devez désigner l'objet que l'utilisateur peut déplacer, de la même manière que vous avez désigné l'objet-cible pour les réponses de type 'objet sensible'. Secundo, il faut désigner la zone de destination, c'est-à-dire la zone dans laquelle l'utilisateur pourra déposer l'objet. Je vous invite à construire ensemble une interaction de ce type.

Exercice 3-6: Venn simple

Réalisons un programme dans lequel l'utilisateur doit placer un disque rouge dans un diagramme de Venn comprenant deux ensembles, les disques, les rouges, et leur intersection.

1. *Premier pas: Créez le décor. Posez une icône d'interaction et dessinez dans cette icône le décor de l'interaction, à savoir les deux ensembles.*
2. *Deuxième pas: Créez l'objet mobile. Posez une icône de présentation au-dessus de l'icône d'interaction et dessinez-y le disque rouge. Nommez cette icône 'disque'*
3. *Troisième pas: Créez la zone de destination 1. Posez une icône de présentation à droite de l'icône d'interaction et choisissez le type de réponse 'target area'. Le dialogue d'édition de cette réponse affiche le message 'move object to target position'. Cliquez sur le disque rouge et déplacez-le dans l'ensemble 'disques'. Authorware confirme votre choix. Primo, il indique en haut le nom de l'icône dans lequel a été créé l'objet que vous avez déplacé ('match object: disque'). Secundo, il positionne la zone de destination à l'endroit où vous avez déposé l'objet. Vous pouvez ensuite ajuster la taille de cette zone et sa position (comme pour les réponses 'zone sensible').*
4. *Faites ensuite de même pour une zone correspondant à l'ensemble des rouges, et une zone pour l'intersection des deux ensembles.*

Pour le type de réponses 'objet mobile'[31], vous pouvez régler un aspect des micro-interactions, le signal de réception de réponse. Authorware propose trois signaux de réception du message: l'objet déposé peut (1) rester à l'endroit où l'utilisateur relâche le bouton de la souris (**'leave at destination'**), (2) être déplacé automatiquement au centre de la zone identifiée (**'snap to center'**) ou (3) être ramené à sa position de départ (**'put back'**). Le signal 2 (centrer l'objet) est intéressant lorsqu'on désire éviter au sujet les tâtonnements liés à un ajustement fin de la position de l'objet. Il peut aussi être interprété comme une confirmation de la réponse de l'utilisateur. Le signal 3 (retour) est utile lorsque le sujet rencontre plusieurs fois la même interaction. Ce signal est aussi souvent utilisé pour ses connotations négatives, par exemple lorsque le sujet pose l'objet en dehors des zones logiques de réponse.

31. A propos de Design : L'objet mobile peut être un texte, ce qui permet par exemple de programmer différentes activités de construction de phrases.

Exercice 3-7: Puzzle

Dessinez un puzzle simplifié, dans lequel il manque une pièce. Présentez trois pièces possibles pour compléter le puzzle et demandez au sujet de déplacer la bonne pièce. Si la pièce correcte est déposée plus ou moins au bon endroit, ajustez sa position avec précision. Si une pièce incorrecte est choisie, remettez-la à sa place de départ.

Si l'utilisateur peut déplacer *n* objets et qu'il y a *m* zones possibles, vous devrez construire *n X m* réponses. Si dans l'exercice 3-6 le sujet avait dû déplacer trois objets (un disque rouge, un disque bleu et un carré rouge) dans quatre zones (les rouges, les disques, l'intersection et en dehors des deux ensembles), vous auriez dû créer 12 réponses possibles. Vous pouvez réduire le nombre de réponses. Si l'utilisateur laisse tomber l'objet en dehors des deux ensembles, peu importe de quel objet il s'agit. Créez donc pour cela une zone qui couvre tout l'écran et choisissez l'option '**match any object**' dans la fenêtre d'édition de cette réponse[32]. La zone est alors disponible pour tout objet que l'utilisateur y déplace.

Exercice 3-8: Venn multiple

Modifier le programme 3.6 et le sauver sous 3.8. Prendre le même décor. Afficher trois objets: un disque rouge, un disque bleu et un carré rouge et demander au sujet déplacer ces différents objets. Si l'utilisateur positionne l'objet en dehors des deux ensembles, il faut le remettre à sa position de départ. Sinon, le laisser à l'endroit où la déposé l'utilisateur. Ajouter un bouton finqui sort de l'interaction lorsque tout est ok.

L'objet que l'utilisateur déplace pendant ces interactions est unique. Or, dans diverses situations, l'auteur aimerait que seul un duplicata de l'objet soit déplacé, de telle sorte que l'utilisateur puisse déplacer plusieurs copies du même objet. C'est par exemple le cas lorsque l'auteur fournit un certains nombre d'éléments de base que l'utilisateur peut assembler en un schéma complexe. Malheureusement, Authorware ne dispose pas d'une option permettant cela. L'astuce pour réaliser néanmoins ce type d'interaction consiste à créer dès le départ de multiples copies du même objet et de sélectionner une copie à la fois au moyen d'une icône de décision (nous étudierons ces icônes dans la leçon 6).

3-7 Librairie de curseurs

Dans plusieurs types de réponse, vous pouvez spécifier un changement de forme de curseur lorsque celui-ci passe sur un objet sensible ou une zone sensible[33]. Les curseurs disponibles se trouvent dans la librairie des curseurs (menu 'libraries', option '**cursors...**'). Au départ, celle-ci contient quelques curseurs fréquents: la barre verticale utilisée pour éditer du texte, la croix fine pour le dessin, la main pour désigner ou déplacer un objet et la montre pour signifier à l'utilisateur qu'il doit patienter quelques instants. Vous pouvez créer d'autres curseurs et les importer dans Authorware (option 'add' dans la fenêtre 'cursor library').

Dans la situation ci-dessus, c'est l'action du sujet qui détermine les modifications de forme du curseur. Vous pouvez aussi forcer un changement de forme du curseur en utilisant une fonction du système. Par exemple, si vous devez effecteur des calculs qui prennent plus de 2 ou 3 secondes, utilisez la fonction 'setcursor (5)' en début de calcul pour remplacer le curseur traditionnel par la montre et 'setcursor (1)' en fin de calcul pour reprendre le curseur traditionnel. Nous verrons dans la leçon 6, comment appeler une fonction dans Authorware.

---

32. Si le nombre total de réponses est trop élevé, il est possible de travailler exclusivement avec des zones de destination de type 'match any object' et d'analyser les réponses du sujet en utilisant les variables du système permettant de connaître le nom de l'objet déplacé par le sujet.

33. A propos de Design : Dans de nombreux logiciels utilisant la manipulation directe, l'utilisateur doit sélectionner un opérateur avant d'en fournir les arguments. Par exemple, il choisit un gomme avant de la poser sur les pixels à effacer. Dans ces cas, le changement de forme du curseur est particulièrement utile car il rappelle à l'utilisateur l'opérateur sélectionné au moment où il en choisit les arguments.

Leçon 4. Structure des interactions

4-1 Objectif

Dans cette leçon, vous apprendrez à mettre de l'ordre dans vos différents feed-back et à 'suivre' le processus d'analyse de réponse.

4-2 Limiter le nombre d'essais

Dans les exercices qui précèdent, le nombre de réponses possibles de l'élève est relativement limité. Toutefois, dans les cas comprenant de multiples zones sensibles ou demandant un déplacement d'objet, l'utilisateur doit parfois recommencer de nombreuses fois avant de trouver la réponse pour laquelle vous avez prévu de sortir de l'interaction. Il est alors nécessaire de limiter le nombre d'essais de l'utilisateur. Ceci sera d'autant plus vrai pour les réponses de type 'texte', que nous étudierons dans la leçon suivante, pour lesquelles le nombre de réponses possibles est illimité.

Limiter le nombre d'essais consiste à construire un feed-back particulier qui se déclenchera au énième essai. Collez une icône de présentation comme pour construire toute autre réponse et choisissez le type de réponse '**tries limit**'. En éditant cette réponse, Authorware vous demande de fixer le nombre d'essais auquel ce feed-back sera déclenché. En général, on présente alors la réponse attendue et on sort de l'interaction. Prenez déjà l'habitude de placer cette icône juste après (= à droite de) la bonne réponse de telle sorte que si l'utilisateur fournit la réponse correcte au énième essai, n étant la limite prévue, il reçoive malgré tout le feed-back associé à la bonne réponse et non celui associé au nombre d'essais.

Exercice 4-1: Droites perpendiculaires (suite)

Reprenez l'exercice 3-5 et sauvez-le sous 4.1. Ajoutez une zone sensible qui couvre tout l'écran pour le cas où l'utilisateur clique en dehors des deux droites. Dans ce cas donner le feed-back "Veuillez cliquer sur le segment B ou le segment C." Donnez la réponse correcte au troisième essai.

4-3 Limiter le temps de réponse

Le type de réponse 'tries limit' vous garantit que l'utilisateur ne reste pas indéfiniment bloqué dans une interaction, pour autant que celui-ci effectue des tentatives de réponses. Par contre, si l'utilisateur ne comprend pas ou ne maîtrise pas la modalité de réponse, il peut simplement attendre indéfiniment que quelque chose se passe. Ce risque est généralement faible parce que Authorware utilise des modes d'interaction largement répandus. Néanmoins, il se peut que la modalité de réponse n'ait pas été communiquée ou pas été perçue et l'utilisateur cherchera par exemple à cliquer sur des objets au lieu de les déplacer, ou encore il les déplacera sans trouver aucune zone sensible. Bref, avec un peu de malchance, il n'effectuera même pas un seul essai et n'atteindra donc pas le nombre d'essais maximum que vous auriez spécifié. Si vous jugez que ce risque existe, vous pouvez utiliser une limitation du temps de réponse.

Un temps maximal de réponse se construit dans Authorware comme une réponse. Construisez une réponse en plaçant une icône de présentation dans la boucle d'interaction et choisissez le type de réponse '**time limit**'. L'édition de cette réponse vous permet de déterminer ce temps maximum (exprimé en secondes), mais aussi de dire s'il s'agit du temps écoulé depuis le début de cette interaction (le temps total au cours des différents essais), ou simplement le temps écoulé depuis le dernier essai. Si c'est le temps depuis le dernier essai, choisissez l'option '**Restart for each try**'. Si vous désirez informer l'utilisateur du temps qui lui reste, sélectionnez aussi l'option 'show time remaining' qui affiche le même réveil (compte à rebours) que l'icône de temporisation[34].

Exercice 4-2: Limitation du temps de réponse

Reprenez l'exercice 3.8 et sauvez le sous 4.2. Si le sujet n'a pas répondu après 10 secondes, affichez une brève explication des modalités de réponse. Si le sujet n'a pas répondu correctement au bout de 20 secondes, donnez lui la réponse correcte (affichez la position attendue des pièces)

Limiter le temps de réponse n'est pas seulement utile pour prévoir des dysfonctionnements de l'interaction. Cette option est utile dans de nombreuses applications, en particulier les jeux et les tests[35].

4-4 Analyse de réponse: comprendre l'ordre des tests

L'analyse de réponse comprend des éléments tels qu'une limite de temps ou une limite du nombre d'essais, éléments qui ne constituent pas des réponses à proprement parler, mais plus exactement certaines vérifications de l'activité de l'utilisateur. En informatique, on parle de tests plutôt que de vérifications. En fait, l'analyse de réponse, c'est-à-dire tout ce qui est positionné sur l'axe horizontal à droite de l'icône d'interaction, n'est rien d'autre qu'une séquence de tests. A chaque point, Authorware fait un test, si ce test est réussi, il descend la branche sous le test, sinon il continue horizontalement et passe au test suivant. Bien comprendre l'enchaînement des tests est particulièrement important dans le cas où les différents tests ne sont pas mutuellement exclusifs.

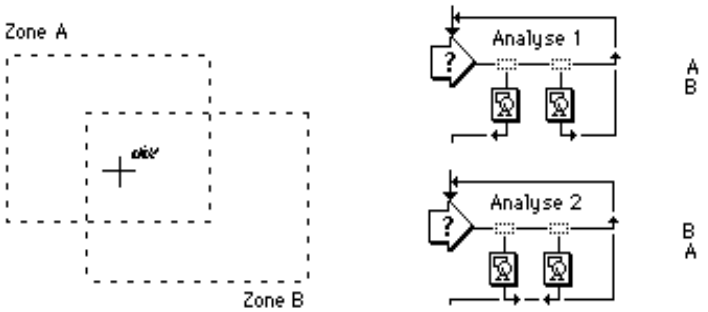


Figure 4-1:A gauche, zone sensibles non-disjointes; à droite, deux analyses de réponse

Prenons un exemple, illustré par la figure 4-1, qui présente un cas que nous avons déjà brièvement abordé. Si le sujet clique sur un point qui appartient à deux zones sensibles, quel feed-back sera fourni par ce programme? La solution dépend de l'ordre des tests et des branchements associés. Dans l'analyse 1, le programme commence par tester si le point se trouve dans la zone A. Comme c'est le cas, le feed-back associé à cette zone sera présenté et le programme quittera l'interaction sans tester si le click tombe dans la zone B. Inversement, dans l'analyse 2, le programme trouvera que le click tombe sans la zone B et recommencera l'interaction sans vérifier si ce point appartient aussi à la zone A. (Ceci s'applique aussi aux autres types de réponse, en particulier aux réponses 'texte' et 'condition' que nous étudierons bientôt).

Dans les cas simples, une séquence de tests ressemble à une **succession de filtres**: seules les réponses qui passent à travers le premier filtre (le résultat du test 1 est négatif) sont confrontées au test 2; seules celles qui ont échoué au 1 et au 2 sont présentées au test 3, etc.

En général, si votre programme ne se comporte pas comme vous pensiez qu'il aurait dû se comporter, met-

34. Le dialogue d'édition vous permet en outre de déterminer ce qui se passe si le compte à rebours est interrompu par une autre activité, par exemple si le sujet utilise les menus (option 'if interrupted by perpetual interaction'). Nous expliquerons ce paramètre quand nous étudierons les interactions perpétuelles (Leçon 10).

35. A propos de Design : Dans les logiciels éducatifs par contre, vous ferez attention aux effets de stress induits par ces contraintes: il arrive que certains utilisateurs regardent paralysés le réveil qui tourne au lieu de tenter de répondre !

tez-vous à sa place et considérez un par un chaque test. Selon les résultats du test, suivez les flèches.

Exercice 4-3: Cible

Créez une cible composée de quatre rectangles concentriques. L'utilisateur clique sur cette cible et le programme affiche le nombre de points: 100, 50, 20 et 10 pour les plages respectives du centre vers la périphérie. Fixer le temps de réponse à 3 secondes par essai. Si le délai est écoulé, afficher 0 points. Dans tous les cas, on recommence le jeu.

Jusqu'à présent, nous avons envisagé deux types de branchement après le feed-back: sortir de l'interaction ou recommencer. Un troisième branchement '**continue**' consiste à reprendre la succession des tests. Ainsi si on reprend le cas de la figure 4-1, et qu'on applique l'analyse 3 illustrée dans la figure 4-2, le sujet recevra le feed-back propre à A, puis le test B étant aussi satisfait, il recevra également le feed-back lié à B.

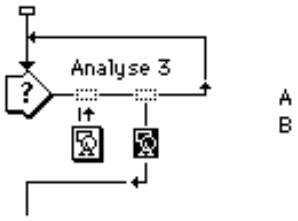


Figure 4-2:Utilisation du mode de branchement 'continue' pour la zone sensible A:

Exercice 4-4: Désignation d'objets

Créez un diagramme de Venn comprenant deux ensembles, les fleurs et les jaunes, dans lesquels le sujet doit déplacer un point noir. Le programme doit deviner l'objet selon l'endroit où le sujet le positionne: il affichera respectivement 'C'est une fleur', 'C'est jaune' ou les deux messages si le point est placé dans l'intersection.

Le mode de fonctionnement 'continue' sera utilisée fréquemment pour réaliser des tests dont les résultats sont stockés provisoirement dans des variables afin d'être utilisés dans un test ultérieur (par exemple, pour le traitement des réponses numériques - cf. leçon 9)

4-5 Analyse de réponse: une recette

La section précédente vous explique comment suivre le fonctionnement, pas à pas, d'une analyse. Celui-ci est valable dans toutes les interactions. En réalité, dans un grand nombre d'interactions, ce raisonnement vous amènera à classer vos réponses selon l'ordre que nous décrivons ici. En général, on retrouve en effet quelques catégories de feed-back communes:

- Les réponses correctes donnent généralement lieu à un **feed-back de confirmation**. Le rôle de ce feed-back a été largement mis en évidence par les théories du renforcement. Selon les cas, ce feed-back peut se limiter à un son, ou au passage à la question suivante. A l'opposé, le feed-back peut fournir des informations complémentaires: "Très bien. C'est effectivement Eddy Merckx qui remporta le tour cette année-là. Il avait remporté le Giro quelques semaines auparavant."
- Les **réponses quasi-correctes** sont généralement assimilées aux précédentes. Il est cependant parfois utile de nuancer le feed-back de confirmation: "En effet, le Mont Blanc approche les 5000 mètres. Son altitude exacte est 4807 mètres." ou "En effet, c'est bien de l'hydrogène. Veuillez noter l'orthographe exacte"[36].
- Les erreurs prévisibles donnent généralement lieu à des **feed-back d'erreur spécifique**. Ces messages sont plus efficaces que des messages génériques du type 'réponse erronée' ou 'syntaxe incorrecte'[37]. La conception de ces feed-back repose sur votre capacité à anticiper les erreurs les plus fréquentes des utilisateurs. Dans les logiciels éducatifs, ces feed-back portent aussi le nom de feed-back de remédiation.
- Les autres réponses erronées sont généralement traitées par un **feed-back d'erreur générique** tel que 'Votre réponse n'est pas correcte. Veuillez recommencer'. Le test associé qui permet d'identifier cette réponse est généralement est une tautologie, c'est-à-dire un test positif quelle que soit la réponse du sujet, par exemple une zone sensible couvrant tout l'écran (parce que les autres réponses auront été filtrées par des tests antérieurs).
- Enfin, lorsque l'utilisateur a effectué un certain nombre de tentatives de réponses ou passé un certain temps dans l'interaction, bref lorsqu'il atteint le critère de fin d'interaction, on présente généralement un feed-back de fin d'interaction tel que "Vous avez commis n erreurs" dans une situation de jeu ou de test ou "Non, la bonne réponse était ..." dans un logiciel éducatif.

Chacune de ces catégories peut être traitée de façon plus nuancée. Le nombre maximal d'essais permet non seulement de mettre fin à l'interaction, mais il peut aussi être utilisé pour fournir des indices de plus en plus précis ou pour nuancer le feed-back de confirmation selon que la réponse correcte ait été fournie au premier essai ou ultérieurement ("Cette fois, votre réponse est correcte").

Nous recommandons de construire l'algorithme d'interaction en considérant dans l'ordre:

1. les **réponses correctes** ou quasi correctes, lesquelles donnent lieu à un feed-back de confirmation;
2. le **critère de sortie**, tel que le nombre maximum d'essais, le temps maximum, ou toute autre limite semblable;
3. les **réponses erronées prévisibles**, lesquelles donnent lieu à un feed-back d'erreur spécifique;
4. les **autres réponses**, lesquelles donnent lieu à un feed-back d'erreur générique (ou de reformulation - cfr. leçon suivante).

La structure décrite ci-dessus couvre un certain nombre de situations interactives, en particulier des situations didactiques, c'est-à-dire des situations dans lesquelles les réponses sont évaluées au sens pédagogique du terme. Mais cette structure simple ne couvre évidemment pas toutes les situations interactives. **Notre expérience nous indique simplement que cette recette guide efficacement le concepteur novice de didacticiels**. Nous étudierons dans la leçon 10 des formes d'interaction qui s'éloignent de cette structure canonique.

---

36. A propos de Design : Ce type de feed-back est notamment important lorsque le concepteur accepte des commandes incorrectes, mais désire que l'utilisateur assimile progressivement les commandes correctes afin d'éviter les cas ambigus.

37. Schneiderman, 1982

Exercice 4-5: Canon à électrons

Demandez à l'utilisateur de cliquer sur le canon à électrons dans le schéma d'un écran de télévision (Cfr cours 'Interaction Personne Machine). Prévoir un feed-back de confirmation, deux feed-back d'erreur spécifique, un feed-back d'erreur générique et fixer le critère de sortie au quatrième essai.

Leçon 5. Les réponses 'texte'

5-1 Objectif

L'objectif de cette leçon est de construire une question au cours de laquelle l'utilisateur répond en entrant du texte au clavier. La principale difficulté consiste à analyser le texte introduit par l'utilisateur

5-2 Principes de base

L'objectif d'un analyseur de réponses ('pattern matcher') est de définir de façon synthétique une classe de réponses 'texte' considérées comme équivalentes par l'auteur. Les solutions les plus avancées permettent de définir la structure syntaxique des phrases à analyser. Des procédures plus simples de pattern matching sont disponibles dans la plupart des langages-auteur, dont Authorware. Ces méthodes reposent sur l'identification d'un ou plusieurs mots-clé. L'auteur peut autoriser des variations orthographiques simples et spécifier quelques paramètres qui influencent la procédure de pattern matching. La procédure de pattern matching consiste en une **mise en correspondance, mot par mot, caractère par caractère, de la réponse de l'utilisateur et du 'pattern' spécifié par le concepteur**. Elle peut se visualiser par le défilement du pattern le long de la réponse analysée (figure 5-1). La fenêtre du pattern 'saute' de mot en mot, jusqu'au moment où un mot de la réponse correspond au pattern. Dans ce cas, le résultat du test est positif.

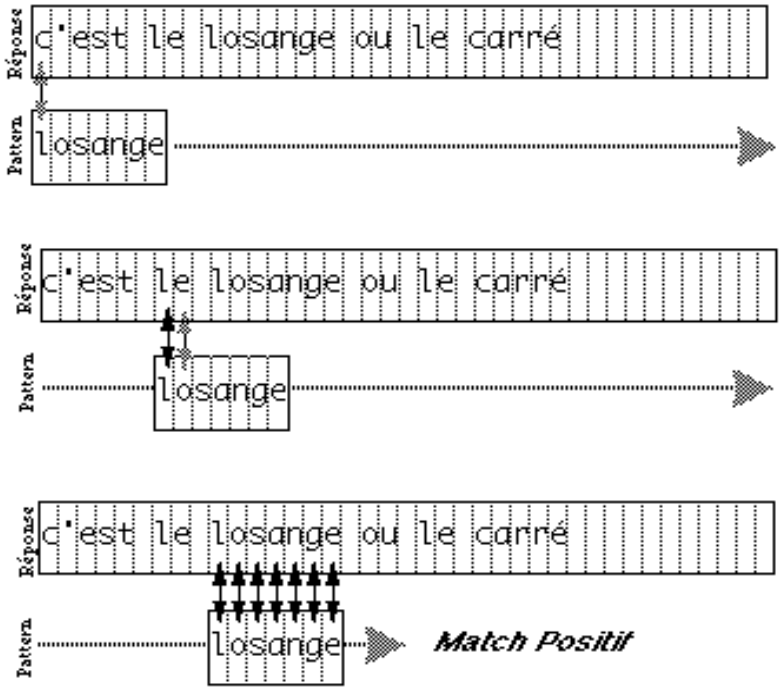


Figure 5-1:Visualisation du fonctionnement de pattern matching

Un pattern est composé de caractères simples et de caractères spéciaux. Un caractère simple doit être mis en correspondance avec un caractère identique dans la réponse analysée. Un caractère spécial peut être mis en correspondance avec plusieurs caractères de la réponse analysée. Les caractères spéciaux disponibles dans Authorware sont présentés dans la section suivante. Si la fenêtre du pattern parcourt toute la réponse sans que ce pattern ait été mis en correspondance avec un élément de cette réponse, le résultat du matching

est négatif (voir figure 5-2)[38].

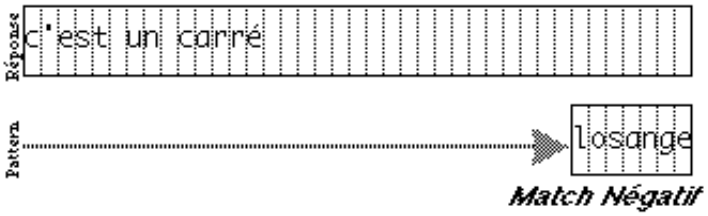


Figure 5-2:La résultat du processus de matching est négatif

Une réponse de type texte se construit comme les interactions précédentes. Sélectionnez le type de réponse **'text entry'**. Dans la fenêtre d'édition, dans le champs **'match if user enters'**, introduisez le mot (pattern) que vous désirez indentifier dans la réponse du sujet.

Exercice 5-1: Le port d'Amsterdam

Poser la question "Par qui est chanté le Port d'Amsterdam?". Fournir un feed-back spécifique pour les réponses "Brel", "Bécaud" et "Bowie".

5-3 Paramétrage de l'analyseur

Vous pouvez modifier différents paramètres de la procédure qui compare votre réponse-cible (le pattern) et la réponse de l'utilisateur.

- Elimination des signes de ponctuation
  Cette option (option '**Ignore: extra punctuation**' dans le dialogue 'Text Response Options') élimine de la réponse tout signe de ponctuation qui ne figure pas dans le pattern. Par contre, si le pattern comporte par exemple un ";", et que la réponse comporte également un ";", celui-ci ne sera pas supprimé afin de permettre le matching ultérieur.
- Elimination des espaces
  Cette option (option '**Ignore: all spaces**' dans le dialogue 'Text Response Options') élimine tous les blancs de la réponse de l'élève. Cette option doit être utilisée avec prudence vu que ce sont les blancs qui décomposent la réponse en mots, lesquels mots sont comparés au pattern. Cette option peut s'avérer utile par exemple dans le cas où le sujet doit répondre par un seul nombre. S'il introduit '1 237' avec un espace après le chiffre des milliers, Authorware identifiera deux nombres dans sa réponse (Cf leçon 9)
- Comparaison de majuscules et minuscules
  Au cours du matching, les lettres du pattern et de la réponse sont comparées deux par deux. Même s'il s'agit de deux lettres identiques, cette comparaison échoue, lorsque l'une est en majuscule et l'autre en minuscule. Le concepteur peut lever cette contrainte au moyen de l'option '**Ignore: capitalization**'.
- Mots excédentaires dans la réponses analysée
  Les explications fournies jusqu'à présent fonctionnent selon le principe du défilement de la fenêtre 'pattern' sur la réponse analysée. Toutefois, ce défilement peut être inhibé en 'dé-sélectionnant' l'option '**Ignore: extra words**' dans le dialogue 'Text Response Options'. Dans ce cas, le pattern et la réponse sont mis en correspondance dès le premier mot de la réponse sans possibilité d'ajustement. Si la réponse du sujet comprend un autre mot que le mot recherché, le matching échoue (Cf. figure 5-3).

38. Le terme 'résultat négatif' n'indique pas si une réponse est correcte ou incorrecte, il indique simplement si le pattern a été identifié. Si le pattern décrit une réponse incorrecte et que la réponse fournie satisfait ce pattern, le résultat de l'analyse sera positif. Les termes 'résultat positif' et 'résultat négatif' signifient respectivement 'pattern identifié' et 'pattern non identifié', ils n'ont pas de signification pédagogique.

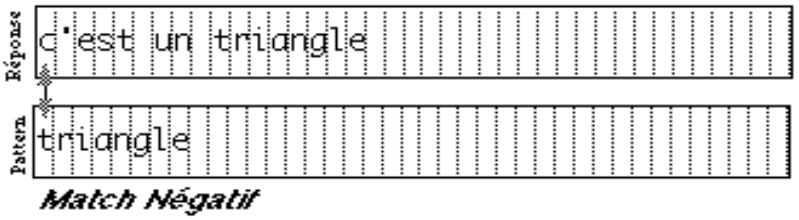


Figure 5-3:Matching lorsque le concepteur n'accepte pas de mot excédentaire

Exercice 5-2: Matching des majuscules et minuscules

Modifier le programme 5.1. (et sauvez le sous 5.2) de telle sorte que celui-ci n'accepte plus les réponses qui ne commencent pas par une majuscule.

5-4 Définition du pattern

Par défaut, le matching n'autorise aucune différence entre les caractères du pattern et de la réponse de l'utilisateur. Toute tolérance dans la mise en correspondance doit être spécifiée au moyen de caractères spéciaux (aussi appelés 'joker'). Authorware offre deux caractères spéciaux[39]: * et ?.

- Le caractère spécial **?** peut être mis en correspondance avec n'importe quel caractère. Il ne permet cependant pas l'oubli de ce caractère, ni l'ajout d'un caractère car ces derniers introduiraient un décalage entre la réponse analysée et le pattern. Par exemple, le pattern ‘losange’ sera identifié dans les mots ‘losange’, ‘lozange’, ‘lorange’... mais pas dans le mot ‘lossange’. (Voir figure 5-4)

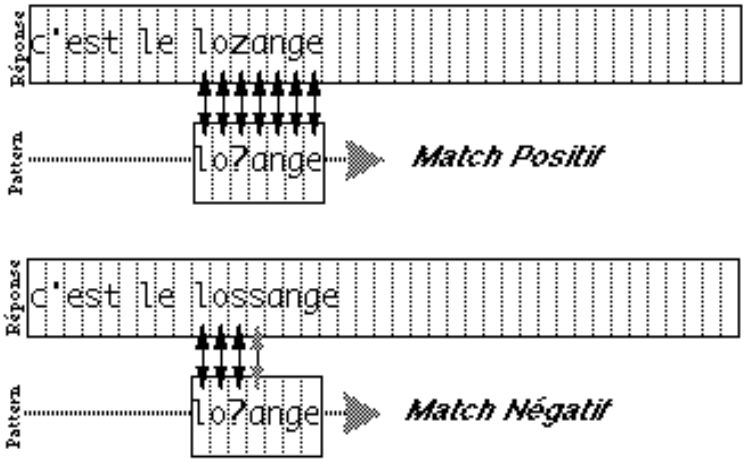


Figure 5-4:Fonctionnement du caractère spécial '?'

- Le caractère spécial * peut être mis en correspondance avec zéro, un ou plusieurs caractères quelconques. Il permet en particulier d'accepter l'oubli d’une lettre, le redoublement incorrect de consonnes, etc. Ainsi le pattern 'lo*sange' décrit en figure 5-5 sera satisfait par les réponses suivantes: losange, lozange, lossange, ... mais aussi ... loripolutange. Par contre, le * n'accepte pas un espace, la réponse 'hello belle orange' ne satisfera pas le pattern. **Si le pattern se constitue uniquement d’un *, il est identifié dans toute réponse comprenant au moins un caractère.**

39. Un caractère spécial peut être déclaré normal (si vous désirez par exemple vérifier que l'utilisateur met un ? au bout de sa question), en le faisant précéder d'un 'backslash' (AltGr + <)

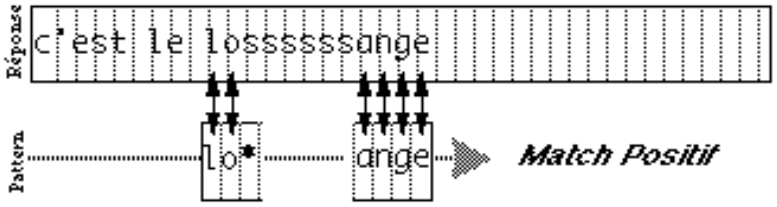


Figure 5-5:Fonctionnement du caractère spécial '*'

Exercice 5-3: Jokers dans les réponses texte

Posez la question: "Entre les étoiles se forment des nuages, appelés nuages interstellaires. Ils sont composés à 90% d'un gaz. Lequel?" prévoir les réponses suivantes. N'offrir qu'un seul essai (c'est-à-dire sortir de l'interaction même pour les réponses erronées).

Réponses	*Feed-back*
hydrogène, hydrogene	*"C'est exact. Ces nuages comportent également des atomes neutres, des ions et quelques molécules."*
ydrogène, hidrogène, ydrogene, ...	*"C'est exact. Note cependant que l'orthographe exacte est hydrogène."*
autres réponses	*"Non, il s'agissait d'hydrogène."*

5-5 Feedback d'erreur générique

La structure des interactions décrites dans la leçon précédente s'applique bien au cas des réponses de type 'texte'. La présence d'un critère de fin d'interaction est très importante pour éviter que l'utilisateur tourne en rond. Le test tautologique (pattern *) identifie toutes les réponses et permettra donc de traiter celles qui n'ont pas été identifiées par les tests antérieurs dans l'interaction. Une certaine prudence est cependant nécessaire dans le feed-back associé à ce test. En effet, si la réponse du sujet ne correspond à aucune des bonnes réponses et à aucune des mauvaises réponses, deux cas sont possibles. Si vous pouvez garantir que toute autre réponse est une réponse erronée, il convient de fournir un feed-back d'erreur générique: "Ta réponse n'est pas correcte." Toutefois, dans le cas des réponses de type 'texte', il n'est pas toujours possible de déterminer qu'une réponse non-identifiée soit incorrecte. Dans ce cas, il est préférable de présenter un **feed-back de reformulation** (moins catégorique): "Je n'ai pas compris votre réponse. Veuillez la formuler plus simplement."

5-6 Patterns conjonctifs

Lorsque l'auteur désire identifier une réponse comprenant plusieurs mots, il suffit d'inclure ces mots dans le pattern en les séparant par un espace: le pattern 'A B' signifie 'trouver A et trouver B'. Vous pouvez en outre spécifier si ces deux éléments A et B doivent être présents dans cet ordre dans la réponse de l'utilisateur. L'ordre n'est pas vérifié si l'option **'Ignore: word order'** dans la dialogue 'Text Response Options' est sélectionnée. Si le concepteur choisit d'ignorer l'ordre des mots, chaque composante du pattern conjonctif est recherchée dans l'ensemble de la réponse. Cela peut se visualiser en imaginant que la fenêtre du pattern revient en début de réponse comme le chariot d'une machine à écrire (figure 5-6). A l'inverse, la procédure de respect de l'ordre peut se visualiser de la manière suivante: lorsque le premier pattern est identifié, la fenêtre continue sa course avec le second pattern, et ainsi de suite pour chaque élément du pattern (figure 5-

7). Si un des patterns n'est pas trouvé, le résultat est négatif.

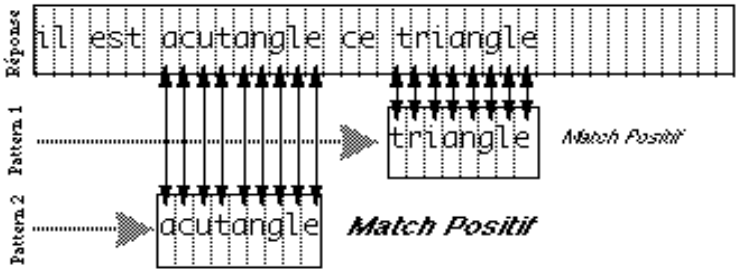


Figure 5-6:Match conjonctif sans respect de l'ordre

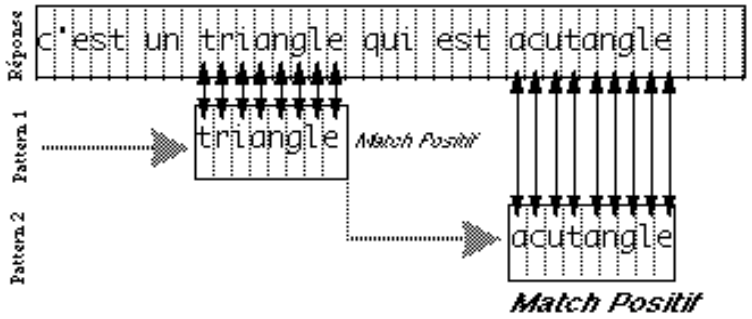


Figure 5-7:Matching d'un pattern conjonctif avec respect de l'ordre

Exercice 5-4: Ordre des mots dans les réponses texte

Demandez au sujet de composer une phrase avec les quatre mots "Lulu" "s'appelle" "chien" et "mon". Prévoir un feed-back positif si les éléments sont fournis dans l'ordre correct. Si les éléments sont fournis dans un autre ordre, demander à l'utilisateur de les entrer dans un autre ordre. Enfin, si un des quatre mots n'est pas présent, indiquez que les mots n'ont pas été correctement introduits et inviter l'utilisateur à recommencer. A la quatrième erreur, donnez la bonne réponse.

5-7 Patterns disjonctifs

Nous avons vu que grâce aux jokers, vous pouvez accepter différentes variations du même mot comme étant synonymes (par exemple 'p?rogravure' considère comme synonyme 'pyrogravure', 'pirogravure', ...). Parfois, vous aimeriez traiter comme synonymes des réponses qui ont peu de caractères en commun, l'utilisateur pouvant par exemple répondre 'Napoléon' ou 'Bonaparte'. Pour définir un pattern disjonctif, le OU s'exprime par une barre verticale (|)[40]. Le résultat de l'analyse sera positif lorsqu'un des patterns spécifiés est identifié dans la réponse. Un processus de matching disjonctif du pattern 'losange|carré' est illustré par la figure 4.7: arrivé en bout de course, la fenêtre reprend la comparaison au début avec un autre pattern.

40. Vous obtenez un | sur PC en pressant la touche 'AltGr' en même temps que le 7.

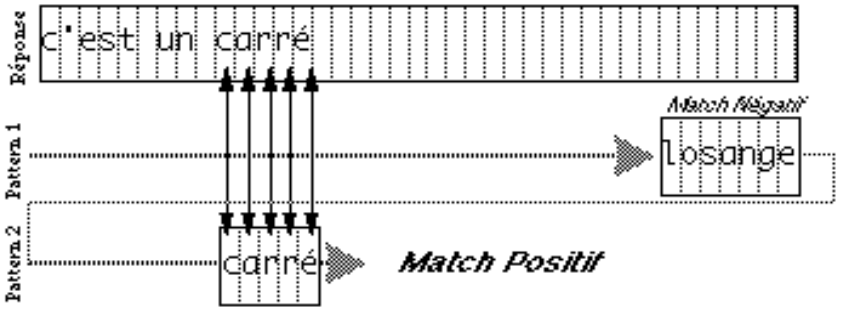


Figure 5-8:Matching disjonctif

Exercice 5-5: Faux-ami anglais

Demandez au sujet quelle est la traduction de 'actually'. Prévoyez comme bonne réponse 'en fait' et 'en réalité'. Si l'utilisateur répond 'actuellement' ou 'maintenant', donnez le feed-back : "Vous vous êtes laissé piéger: 'Actually' signifie 'en réalité'." Prévoyez un feed-back pour toute autre réponse incorrecte. Donnez la bonne réponse, au troisième essai.

5-8 Patterns complexes

Le pattern 'A B | C D' correspond à la formule '(A et B) ou (C et D)'. Les réponses 'A B' et 'C D' seront acceptées. Si les contraintes d'ordre sont levées (voir ci-dessous), les réponses et 'B A' et 'D C' seront également acceptées. Par contre, les réponses 'A D', 'A C', 'B C', ... conduiront à un résultat négatif. Aussi, si vous désirez accepter deux éléments parmi quatre, vous devriez spécifier toutes combinaisons possibles 'A B', 'A C', 'A D', 'B C', 'B D' et 'C D'. Si vous désirez accepter trois éléments, cela donne 10 combinaisons à prévoir et ainsi de suite. Authorware vous permet d'accepter une satisfaction partielle du pattern complet: vous spécifiez un pattern complet, ensuite dans le champs **'match at least ? words'** vous indiquez le nombre de mots qui doivent être identifiés[41].

Exercice 5-6: Patterns complexes

Demandez au sujet de citer trois villes de Suisse romande. Prévoir les réponses suivantes: Genève, Lausanne, Fribourg, Neuchâtel, Sion, Martigny. Si aucune ville correcte n'est fournie, donnez le feed-back "Je ne trouve pas de ville romande dans votre réponse. Essayez encore.". Si moins de trois villes romandes sont communiquées, donnez le feed-back "Votre réponse comprend moins de trois villes romandes. Recommencez."

Vous constatez que dans l'exercice 5-6., si l'utilisateur introduit 2 villes au premier essai, il ne peut se contenter d'entrer la ville manquante au second essai, il doit toutes les entrer à nouveau. Vous éviter cela sélectionnez l'option 'incrémental matching': ainsi pour chaque réponse dans laquelle il doit identifier *n* mots, Authorware mémorise les mots déjà identifiés au cours des essais précédents.

41. Un test sur le pattern 'A B C D', avec l'option 'match a least 1 word', est équivalent au test sur le pattern disjonctif 'A | B | C | D'.

Exercice 5-7: Patterns complexes (suite)

Modifiez l'exercice 5-6 de telle sorte que, en cas de réponse incomplète, le sujet doive seulement communiquer les villes manquantes. Sauvez-le sous 5.7.

5-9 Analyses négatives

Authorware ne permet pas d'exprimer un pattern du genre 'accepter tel mot si tel autre mot n'est pas présent'. Le pattern peut comprendre des conjonctions et disjonctions mais pas de négation. Toutefois, vous pouvez arriver à un résultat équivalent en plaçant en début d'analyse un pattern qui identifie les mots indésirables et exclut immédiatement ces réponses du reste de l'analyse.

Exercice 5-8: Analyses négatives

Posez la question "A quelle espèce appartient la baleine?" et traitez deux réponses 'poisson' et 'mammifère'. Toutefois, il faudrait éviter que l'utilisateur puisse taper "ce n'est pas un poisson" ou "ce n'est pas un mammifère". Dans ces deux cas demandez-lui d'éviter les formulations négatives.

5-10Traitement du langage naturel

Exercice 5-9: Traduction anglais-français

Demandez au sujet de traduire "My brother's dog has bitten the policeman". Donnez un feed-back pour les réponses qui vous semblent correctes. Dans les autres cas, demandez de recommencer. Donnez la bonne réponse au troisième essai.

Si le pattern recherché est une phrase, vous pouvez soit exiger un respect strict de l'ordre des mots du pattern, soit accorder une liberté totale. Malheureusement, il n'existe pas de niveau de contrainte intermédiaire basé sur la syntaxe de la langue. De même au niveau des mots excédentaires, vous pouvez les accepter ou les exclure tous, mais vous ne pouvez pas dire 'exclure tous les substantifs, mais accepter les articles et les prépositions'. Il n'y a donc pas de bonne solution au problème 5-9:

- Si vous définissez la réponse exacte comme pattern, avec ordre strict, la phrase 'Le chien qui appartient à mon frère a mordu le policier' sera rejetée.
- Si vous définissez le pattern 'chien frère mordu policier', avec ordre strict, la phrase 'Le chien de mon frère a chassé celui qui a mordu le policier' serait acceptée bien qu'incorrecte, alors que la phrase 'Le policier a été mordu par le chien de mon frère' serait rejetée bien que correcte.
- Si vous définissez le pattern 'chien frère mordu policier', sans contrainte d'ordre, la phrase "Mon frère a mordu le chien du policier' serait acceptée bien qu'incorrecte.

Bien que l'analyseur de réponses 'texte' soit un des atouts d'Authorware par rapport aux autres outils de développement d'applications interactives, il s'agit encore d'un traitement assez superficiel des réponses, basé sur l'identification de mots et non de structures. Certains auteurs développent une grande ingéniosité pour construire les patterns adaptés aux situations. Mais, il faut accepter que toute analyse reposant simplement sur des mots-clés est limitée. Vous trouverez dans les cours quelques explications sur l'analyse du langage naturel au moyen des techniques d'intelligence artificielle. En attendant que les langages-auteur comprennent des analyseurs capables des mêmes performances, vous devez simplement tenir compte de ces contraintes techniques. Soyez pragmatiques: votre but n'est pas de faire du traitement de la langue naturelle pour le plaisir, mais de pouvoir discriminer différentes réponses, par exemple parce qu'elles attestent de différents niveaux de connaissances. L'art du concepteur est dès lors de créer des situations dans lesquelles les niveaux de connaissance que vous voulez discriminer produisent des énoncés qu'Authorware pourra discriminer.

5-11Micro-interactions

Comme pour les types de réponse étudiés dans les leçons précédentes, vous pouvez régler un certain nombre de paramètres qui déterminent le déroulement des micro-interactions. Toutefois, ces paramètres ne sont pas édités dans l'icône qui représente la réponse comme c'était le cas jusqu'ici, mais dans l'icône même d'interaction, sous l'option '**text entry options**'. En effet, quel que soit le nombre de tests effectués sur la réponse textuelle du sujet, Authorware ne permet qu'une zone de réponse par icône d'interaction. Aussi, les paramètres spécifiés sous 'text entry option' seront valables pour toutes les réponses 'texte' d'une même interaction.

- Le sujet est implicitement informé de la modalité de réponse par l'apparition de la zone de la réponse et la présence d'un 'prompt' en début de zone-réponse (un petit triangle qui clignote). Vous pouvez supprimer le 'prompt' en de-sélectionnant l'option '**show entry marker**', notamment lorsque ce symbole gêne le graphisme de l'interaction (par exemple, dans les textes lacunaires). Même si la compréhension de ce symbole par l'utilisateur n'est pas garantie, ne le supprimez pas sans raison !
- Vous pouvez régler différentes caractéristiques de la réponse: longueur maximale ('character limit'), police et taille de caractères, position et dimensions de la zone de réponse ('entry area position'). Ces informations ne sont pas neutres: elles informent indirectement le sujet sur la longueur maximale de la réponse et donc sur les réponses possibles.
- Vous pouvez modifier le signal d'émission de la réponse. D'une part, le signal d'émission par défaut est la touche RETURN. Il est parfois intéressant de remplacer ce signal par une autre touche, telle

que ENTER ou une touche-fonction, notamment lorsque la réponse de l'utilisateur doit elle-même pouvoir contenir un RETURN (par exemple si l'utilisateur entre un texte au clavier). D'autre part, vous pouvez supprimer le signal d'émission de réponse: si vous fixez un nombre maximal de n caractères ('character limit'), et sélectionnez en outre l'option 'auto entry', Authorware génère automatique un signal d'émission dès que le sujet a entré le énième caractère. Attention, cette option est très dangereuse car les utilisateurs ont pris l'habitude de presser RETURN et risquent donc de presser return de toute manière, ce signal d'émission excédentaire étant stocké dans le buffer d'entrées[42] et traité comme signal d'émission pour l'interaction suivante.

- Le signal de réception de la réponse est l'inversion de la couleur de fond de la zone de réponse. Vous ne pouvez modifier ce signal. En outre, ce signal a pour effet de sélectionner le texte de la réponse de telle sorte que, lors d'une nouvelle tentative de réponse, le premier caractère introduit par l'utilisateur effacera sa réponse antérieure
- Par défaut, Authorware ignore les réponses vides, c'est-à-dire le cas où le sujet presse RETURN sans avoir introduit le moindre caractère. Cette réponse vide n'est pas analysée, ni comptabilisée dans le nombre d'essais. Elle est simplement ignorée. Or, dans certains cas, il est intéressant de permettre à l'utilisateur de donner une réponse vierge. Dans ce cas, de-sélectionnez l'option 'ignore null entries'.
- Enfin, par défaut, la réponse du sujet est effacée en même temps que les autres éléments de l'interaction en quittant l'icône d'interaction. Si vous désirez laisser cette réponse pour la traiter ultérieurement, de-sélectionnez l'option 'erase entry on exit'.

Exercice 5-10: Texte Lacunaire

Créez un text lacunaire : "Un carré possède quatre côtés ..... et quatre angles .....". Supprimez le 'prompt', harmonisez le style du texte et de la réponse, évitez que la réponse de l'utilisateur recouvre la suite du texte. Veillez à ce que la réponse fournie dans le premier trou ne soit pas effacée au moment d'entrer la réponse pour le second. Prévoir les réponses correctes 'isométriques' ou 'égaux' pour la première réponse; 'droits' ou '90' pour la seconde.

42. La fonction 'FlushKeys' vous permet de vider manuellement le buffer d'entrées.

5-12Réponses de type 'touche'

Une réponse de type 'touche' ('keypress') permet à l'utilisateur de répondre en pressant une seule touche (sans presser RETURN). Ce type de réponse était assez populaire avant l'apparition des interfaces graphiques. En éditant la réponse, il suffit d'indiquer au système (champs '**key**') la touche pressée. Pour accepter une touche quelconque, utilisez le même joker mono-caractère (le ?) que celui utilisé dans l'expression des patterns d'analyse de réponse texte. En fait, une réponse touche est équivalente à une réponse de type 'texte', dans laquelle vous auriez fixé la longueur maximale de réponse à 1 caractère et sélectionné l'option 'auto entry'. Ce type de réponse est surtout utilisé pour créer des système de commandes, simples mais rapides.

Exercice 5-11: Réponses 'touche'

Demandez au sujet de presser la touche b pour bleu, r pour rouge, j pour jaune et v pour vert. Affichez pendant 0.5 secondes une forme géométrique dont la couleur correspond à la touche choisie.

Leçon 6. Variables et fonctions

6-1 Objectif

La plupart des systèmes informatiques maintiennent des informations en mémoire, notamment concernant le comportement de l'utilisateur. Ces informations sont conservées dans ce qu'on appelle des 'variables' et traitées par ce qu'on appelle des 'fonctions'. J'en explique les bases dans ce module. Par expérience, la notion de variable constitue la principale, je dirais même la seule, **difficulté** dans l'apprentissage d'Authorware. Je vous recommande donc la plus grande attention pour les éléments de cette leçon. Par contre, si vous avez déjà appris un autre langage de programmation et que vous êtes familiers avec le concept de variable, vous n'aurez pas de difficulté pour appliquer à ce qui suit vos connaissances préalables. Vous serez simplement attentifs à certaines options d'Authorware et au grand nombre de variables que le système gère pour vous.

6-2 Variables Authorware

Lorsque vous demandez à Authorware de quitter l'interaction au 3ième essai, comment sait-il que le sujet a déjà réalisé trois essais? Tout simplement, parce qu'il garde cette information en mémoire: il la met à jour à chaque tentative de réponse et la compare chaque fois avec le nombre d'essais maximum que vous avez défini. Vous ne devez pas retenir l'adresse exacte de cette information dans la mémoire. Il est plus simple de se représenter la mémoire du système comme une immense armoire avec d'innombrables tiroirs contenant chacun une information précise. Chaque tiroir porte un nom. En particulier, le tiroir dans lequel est inscrit le nombre de tentatives de réponse s'appelle 'tries'. Un autre tiroir contient le nombre de bonnes réponses fournies par l'utilisateur[43], il s'appelle 'total correct'. Le terme 'variable' indique le fait que le contenu du tiroir peut varier en cours d'exécution du programme.

Vous pouvez, en tant qu'auteur, voir le contenu d'un tiroir, d'une variable. Par exemple, exécutez le programme de l'exercice 5-6 et comptez mentalement le nombre d'essais que vous réalisez. Ensuite, sélectionnez l'option **'variables'** dans le menu 'window'. Authorware ouvre une boîte de dialogue en haut de laquelle figure un menu 'category'. Sélectionnez la catégorie 'interaction'. Authorware affiche alors une liste de variables. Sélectionnez la variable 'tries'. Authorware vous affiche le contenu de ce tiroir dans le coin supérieur droit, dans le champs 'current value'.

Authorware gère automatiquement environ 200 variables. Il s'agit d'un des points forts de ce langage en termes de productivité. Ces variables sont classées en 9 catégories[44]. En voici quelques exemples:

Catégorie	Variable	Contenu
File	FileLocation	Chemin d'accès au répertoire dans lequel se trouve le fichier Authorware en cours d'exécution.
Framework	HotTextClicked	Contient de dernier 'hot text' sélectionné par l'utilisateur (voir leçon 11 sur les hypertextes)
Time	FullTime	Heure actuelle en heures, minutes , secondes.
General	ClickX ClickY	Coordonnées du dernier point où l'utilisateur a cliqué.
Interaction	PercentCorrect	Pourcentage de réponses correctes fournies par l'utilisateur depuis le début du programme
Interaction	WordCount	Nombre de mots dans la réponse-texte du sujet

43. Le nombre de réponses correctes est le nombre de fois que l'utilisateur a fourni une des réponses que vous avez vous-même définies comme correctes ('correct answer').

La boîte de dialogue qui présente les variables vous fournit une explication sur leur contenu. La présence de majuscules ou de minuscules dans le nom de variable est uniquement destinée à rendre les noms plus lisibles, étant donné qu'Authorware n'accepte pas la présence d'espaces ou de traits d'union dans un nom de variable. La variable WordCount susmentionnée contient le nombre de mots introduits lors de la dernière interaction. Le nombre de mots introduits dans une interaction antérieure, par exemple à l'icône dénommée 'Question4' est mémorisé sous la variable **WordCount@"Question4".** Le signe @ est utilisé pour les variables dont Authorware conserve la valeur prise à chaque icône du programme. Dans certains cas, vous pouvez modifier le contenu des variables. Nous verrons comment lorsque nous parlerons de la création de variables.

Vous pouvez afficher le contenu d'une variable sur l'écran de l'utilisateur. Imaginons que vous désirez afficher l'heure dans une icône de présentation. Editez cette icône et tapez[45] le texte **{FullTime}**. La présence d'accolades autour du nom de la variable indiquez à Authorware qu'il ne s'agit pas d'afficher simplement les caractères f-u-l-l-t-i-m-e à l'écran, mais d'afficher le contenu d'un tiroir qui s'appelle 'fulltime'.

En principe, Authorware lit le contenu de la variable au moment où vous en demandez l'affichage. Néanmoins, vous pouvez demander de réactualiser l'affichage à chaque fois que la valeur (interne) de la variable change: en éditant l'icône, sélectionnez la variable (comme pour toute sélection de texte), choisissez dans le menu 'Display' de 'Properties' et cochez la check-box **'Update Displayed Variables'**. Soyez cependant attentif au fait que, si les variables affichées changent souvent de valeur (comme c'est le cas dans l'affichage du temps en secondes), l'actualisation permanente de l'affichage ralentira forcément l'exécution du programme.

Exercice 6-1: Phrases chronométrées

Demandez à l'utilisateur d'introduire une phrase au clavier. Acceptez toutes les phrases. Après chaque phrase, afficher le message du type "Votre phrase nº3 contient 5 mots, vous avez mis 15 secondes pour les introduire.". Ensuite, demander à l'utilisateur d'introduire une nouvelle phrase.

6-3 Les fonctions offertes par Authorware

Authorware utilise une interface de programmation graphique qui est très conviviale. Néanmoins, vous avez peut être déjà ressenti certaines limitations de cette forme de programmation. Prenons un exemple. Typiquement, si vous voulez dessiner un rectangle, vous éditez un icône de présentation et tracez simplement le dit rectangle avec l'outil de dessin approprié. Vous êtes cependant obligé de fixer la taille et la couleur de ce rectangle. Dans un langage de programmation classique, vous auriez introduit dans le code une instruction indiquant au logiciel le type de forme à dessiner et ses différents paramètres (position, dimensions, couleur,...). Cette possibilité vous est également offerte. Nous vous la présentons en pas-à-pas.

Exercice 6-2: Fonctions de dessin

Le but est de faire dessiner un rectangle au programme.

Premier pas. Prenez une icône de présentation et dessinez-y un rectangle noir et blanc comme vous l'avez fait jusqu'à présent.

Deuxième pas. Prenez une icône de calcul *.Placez-la sous l'icône de présentation.*

Double-cliquez sur cette icône. Elle s'ouvre sur une page blanche dans laquelle vous allez écrire les lignes de programme. Tapez la ligne suivante:

Box (1, 100, 100, 200, 200)

---

44. Une dixième catégorie, appelée 'all' regroupe toutes les variables, toutes catégories confondues, afin de vous faciliter la recherche au cas où vous ignorez à quelle catégorie appartient une variable particulière.

45. Au lieu d'entrer le nom de la variable en le tapant au clavier, vous pouvez - au moment d'introduire le nom de la variable - sélectionner la variable dans la liste des variables disponibles et cliquer simplement sur le bouton 'paste' pour coller cette variable dans votre programme.

Troisième pas. Fermez le fenêtre de cette icône de calcul. Authorware vous demande de confirmer vos choix, dites OK. En cas de message d'erreur, vérifiez bien si vous n'avez pas commis d'erreur en recopiant la ligne ci-dessus. ***Attention sur certains systèmes, le séparateur*** *(le symbole utilisé pour distinguer deux arguments dans une commande)* ***n'est pas la virgule mais le point-virgule****. Faites un run de votre programme.*

Quatrième pas. L'exécution du programme fait apparaître deux rectangles, celui que vous avez tracé manuellement dans l'icône de présentation et celui que vous avez fait dessiner par Authorware dans l'icône de calcul. Modifiez les paramètres de ce dernier afin que les deux rectangles se superposent exactement. Le premier paramètre (1) détermine l'épaisseur du trait en pixels, les paramètres suivants définissent les coordonnées de la diagonale du rectangle, dans notre cas, du point (100, 100) au point (200, 200). Le système de coordonnées graphiques dans un écran est représenté par la figure 6.1.

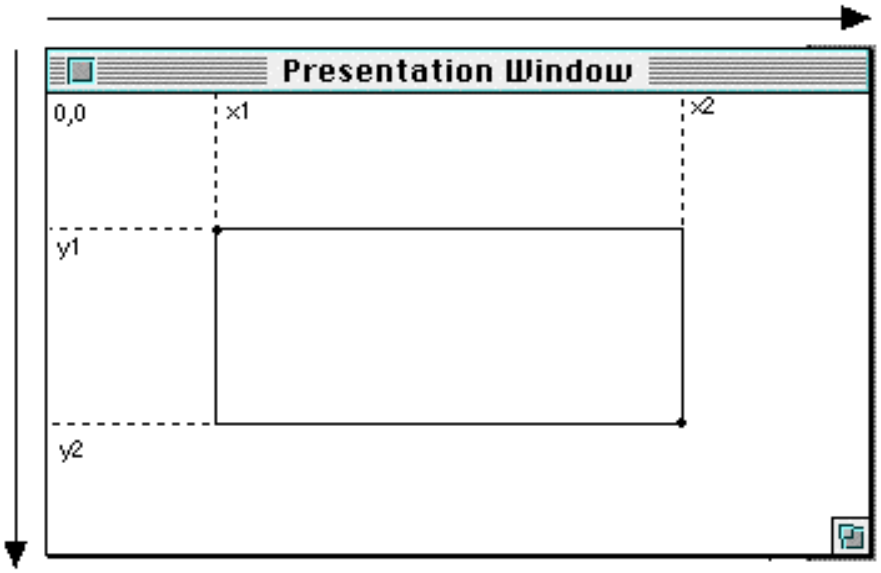


Figure 6-1:Dessin d'un rectangle (bordure noire) au moyen de la fonction box (1;x1; y1; x2; y2). L'origine des axes se trouve dans le coin supérieur gauche de l'écran. (les traitillés ont été ajoutés manuellement pour indiquer le système de coordonnées)

Dans l'exercice 6.1, l'ajustement des paramètres de la fonction pour obtenir exactement le rectangle désiré peut s'avérer fastidieux. Dans ce cas en fait, la programmation graphique est plus rapide que l'utilisation de fonctions. Par contre, une des limitations de la programmation graphique est le manque d'abstraction: vous dessinez un rectangle particulier, dont tous les paramètres sont fixés. Si vous utilisez la fonction 'box' comme ci-dessus, vous pouvez utiliser des variables comme paramètres de la fonction (au lieu des valeurs utilisées précédemment).

Exercice 6-3: Rectangle temporel

Demandez au sujet de cliquer sur un bouton " dessin ". Comme feed-back dessinez un rectangle dont la largeur est proportionnelle au temps de réponse de l'utilisateur. Ensuite, l'utilisateur recommence. Mettre un temps de réponse maximal de 10 secondes.

Comme pour les variables, Authorware met à votre disposition environ 200 fonctions, réparties en différentes catégories: gestion des fichiers, navigation, traitement de chaînes de caractères, fonctions mathématiques, fonctions de base de programmation, etc.

Les arguments d'une fonction (par exemple les coordonnées du rectangle dans la fonction 'box') sont placés entre parenthèses et séparés par un séparateur, la virgule ou le point-virgule selon les réglages de votre système.

Intéressons-nous à une fonction très utile: **l'affectation**. Affecter signifie donner une valeur à une variable ou, selon la métaphore utilisée ci-dessus, placer une information dans un tiroir. On parle de la valeur d'une variable pour désigner son contenu. Le contenu d'un tiroir peut être un nombre (variables numériques), des caractères (variables alphanumériques) ou simplement l'information vrai/faux (variables booléennes). L'affectation s'écrit comme dans les exemples ci-dessous:

```
Prénom := "Albert"   écrire les caractères A-l-b-e-r-t dans le tiroir 'Prénom'
Prix := 167 + 54     Faire la somme de 167 et 54 et mettre le résultat (221) dans
                     le tiroir 'prix'
Prix := Prix * 10    Prendre le contenu du tiroir 'prix' (221), le multiplier (*)
                     par 10 et mettre le résultat (2210) dans le tiroir prix.
```

Il est très important de bien comprendre le dernier exemple. L'opérateur d'affectation(:=) ne correspond pas à 'égal', bien que de nombreux langages utilisent simplement '=' pour exprimer l'affectation. Il faut donc lire l'opérateur 'A:=B' comme '**mettre B dans A**'. Il serait plus correct d'écrire quelque chose comme prix * 10 ⇒ prix', si cette flèche était un caractère aisément accessible sur les claviers. Ici, l'expression se lit dans le sens inverse, de la droite vers la gauche, comme une formule mathématique: 'prix ⇐ prix * 10 '.

J'attire votre attention sur le fait qu'un tiroir ne peut comprendre qu'une valeur et que par conséquent, l'ancienne valeur de la variable 'prix' est perdue, seule la dernière valeur affectée est stockée. Dit autrement, **la nouvelle valeur de la variable écrase la valeur précédente**. Dans de nombreux cas, on augmente progressivement la valeur d'une variable. On parle alors **d'incrémenter** la variable.

Exercice 6-4: Cercle variable

Dessinez un cercle et invitez le sujet à cliquer sur deux boutons: plus grand ou plus petit. Selon sa réponse, augmentez ou diminuez la taille du cercle de 10 points, redessinez le cercle puis invitez l'utilisateur à recommencer.

6-4 Créer vos propres variables

Selon le logiciel que vous développez, vous aurez besoin de conserver d'autres information que celles prévues par Authorware. Ces variables de l'auteur sont appelées **'custom'** par Authorware. Pour créer une nouvelle variable, soit vous sélectionnez l'option **'new variable'** du menu 'data', soit vous notez simplement cette variable dans une icône de calcul. Dans le dernier cas, au moment où vous fermez la fenêtre de calcul, si Authorware identifie une variable inconnue, il ouvre la boîte de dialogue de création de variables. Celle-ci comporte trois champs:

- Le nom ne peut comprendre ni espaces ni traits d'union. Pourtant, je vous recommande vivement d'utiliser un **nom significatif** (par exemple "ResultatTest3") plutôt qu'un symbole anonyme (du genre N ou R). En effet, un logiciel interactif compte rapidement plusieurs dizaines de variables dont l'auteur finit généralement par oublier la signification, en particulier s'il désire modifier ce programme plusieurs mois après sa création. Pour rendre le nom plus lisible, utilisez des minuscules et des majuscules comme dans l'exemple ci-dessus[46].
- La valeur initiale de la variable ('initial value'), c'est-à-dire la valeur qu'Authorware lui affectera lors de la mise en route du programme. Par défaut, la valeur initiale est 0 pour les variables numériques et "" (aucun caractère) pour les variables alphanumériques.
- Vous êtes enfin invité à décrire brièvement la variable créée, pour vous-même ou pour toute autre personne désirant utiliser, comprendre ou modifier son programme ('description'). Perdre quelques secondes pour documenter chaque variable vous fera économiser un temps plus important lorsque vous devrez comprendre le rôle d'une variable créée plusieurs semaines auparavant.

Une fois créées, les variables figurent dans la liste des variables ('show variables') dans le menu 'Window'. Les variables que vous créez se trouvent dans une autre catégorie, qui porte le nom de votre programme[47].

Exercice 6-5: Mini-calculette

Créez une mini-calculette comprenant quatre boutons, 'multiplier par 10', 'diviser par 2', 'ajouter 1' et 'Remettre à zéro'. La calculette affiche toujours le résultat du calcul. Au départ, la calculette affiche 0.

6-5 Format de présentation des nombres

Vous pouvez régler le format d'affichage des variables numériques en sélectionnant l'option **'number format'** dans le menu 'Text'. Les différents réglages sont:

0,5 ou 0.5	La partie décimale des nombres peut être séparée soit par un point soit par une virgule. Vous pouvez aussi laisser Authorware choisir entre ces deux séparateurs selon le système utilisé :option 'use local standard'.
0,76788 ou 0,76	Vous pouvez choisir le nombre de chiffres décimaux à afficher ('Include ? digits to right of decimal'). Vous pouvez aussi décider de n'afficher aucun chiffre décimal en dé-sélectionnant l'option 'show numbers after decimal'.
0,3 ou 0,300	Si vous sélectionnez l'option 'include trailing zeros', Authorware affiche les zéros nécessaires pour atteindre le nombre de chiffres décimaux spécifiés.
2.000 ou 2,000	Les chiffres des milliers et des millions peut être réglé de la même manière que le séparateur des chiffres décimaux.
19 ou 0019	Vous pouvez ajouter des zéros à gauche du nombre de telle sorte que celui-ci

46. Contrairement à d'autres langages et à d'anciennes versions d'Authorware, vous ne devez pas spécifier de quel type de variable il s'agit (numérique, alphanumérique ou booléenne). Vous devez toutefois savoir quel type d'information y est stocké puisque certains opérateurs propres aux variables numériques ne fonctionnent pas sur les variables alphanumériques et vice-versa.

47. Si vous n'avez pas encore sauvé votre programme, celui-ci s'appelle [untitled]; vos variables se trouvant donc dans la catégorie [untitled].

comprenne toujours le même nombre de caractères (option 'Include leading zeros to ? places').

Exercice 6-6: Mini-calculette (suite)

Modifiez le programme 6.5 et sauvez le sous 6.6. A chaque opération de l'utilisateur, celle-ci s'applique à trois variables (= 3 nombres entrés par l'utilisateur) affichées les unes sous les autres. Réglez le format d'affichage de telle sorte que les unités soient toujours alignées verticalement.

6-6 Structure itérative

Dans une icône de calcul, les 'instructions sont exécutées linéairement de la première (en haut) à la dernière (en bas). Il est souvent utile de pouvoir exécuter plusieurs fois une instruction ou un groupe d'instructions. Une **boucle** est une partie du programme qui est exécutée plusieurs fois. Nous nous intéressons ici aux boucles à l'intérieur même de l'icône de calcul. En termes d'algorithme, on parle aussi d'itération

Créer un boucle c'est dire par exemple à Authorware: "Répétez 10 fois telles instructions". Pour cela vous spécifiez primo quelles instructions font partie de la boucle et secundo le nombre de répétitions.

```
repeat with counter := 1 to 10
        beep ()
    end repeat
```

Cette instruction dit de répéter la partie encadrée, c'est-à-dire les lignes situées entre la première ligne ('repeat...') et la dernière ('end repeat'). La ligne 'end repeat' n'a pas d'effet en soi, elle délimite simplement ce qui doit être répété (le corps de la boucle). Dans l'exemple ci-dessus, le corps de la boucle comprend une seule instruction "Beep ()", qui émet un 'bip'. Les parenthèses vides indiquent que cette fonction ne prend pas d'argument.

Les termes 'with counter := 1 to 10' indiquent le nombre de répétitions. Au lieu de dire 'répéter 20 fois', on dit de 'commencer avec une variable qui s'appelle 'counter' et qui vaut 1, ajouter 1 à chaque itération, jusqu'au moment où cette variable vaut 10. Le nom de la variable est arbitraire.

Exercice 6-7: Message d'attente

Créez un programme qui affiche le message 'Je prépare les données'. Affichez le curseur 'montre'. Créez un boucle qui, à chaque passage, dessine un rectangle qui se noircit progressivement comme lors de la copie de fichiers.

Le nombre d'itérations peut être exprimé par un décompte: 'with counter := 10 down to 1'. Vous pouvez aussi fixer la fin de la boucle par une condition. Dans le cas, les instructions sont répétées tant que ('while') la condition est vérifiée.

```
largeur := 10
longueur :=  100
repeat while largeur < longueur
       largeur := largeur + 10
       longueur :=  longueur + 1
       Box (-1; 100;100; 100+longueur;100+largeur)
     end repeat
```

Dans l'exemple ci-dessus, la largeur est initialement inférieure à la longueur, mais, à chaque itération, la première augmente de 10 et la seconde de 1. La boucle est répétée tant que la largeur est inférieure à la longueur. Essayez-le.

Le corps d'une boucle peut contenir de multiples instructions, et notamment une autre boucle. Testez le programme suivant.

```
repeat with N1 := 1 to 30
      repeat with N2 := 1 to 25
          X := N1 * 20
          Y := N2 * 20
          circle (-1, X, Y, X + 10, Y +10)
      end repeat
end repeat
```

Pour chaque valeur de N1, ce programme exécute la boucle concernant N2. Donc, pour N1=1, il considère toutes les valeurs de N2, de 1 à 25. Il passe ensuite à N1=2 et reconsidère toutes les valeurs de N2, de 1 à 25, et ainsi de suite. Comme N1 et N2 sont respectivement utilisés pour calculer les coordonnées horizontales et verticales, les différents cercles sont dessinés colonne par colonne (les coordonnées X et Y sont obtenues en multipliant respectivement N1 et N2 par 20 simplement pour rendre les effets plus visibles à l'écran).

Prenez l'habitude d'indenter vos programmes c'est-à-dire d'aligner verticalement les éléments qui sont associés: le premier 'end repeat' marque la fin de la boucle sur N2 alors que le second 'end repeat' marque la fin de la boucle sur N1. On visualise ainsi mieux l'emboîtement des boucles, un peu comme des poupées russes.

Exercice 6-8: Dessin de cercles

Modifiez le programme ci-dessus de telle sorte que les petits disques apparaissent rangée par rangée au lieu de colonne par colonne.

6-7 Structure conditionnelle

Jusqu'à présent, Authorware exécutait systématiquement chaque instruction de votre programme. Il ne vous était pas possible, si on reprend par exemple l'exercice 6.8, d'afficher alternativement un disque et un carré, car cela implique que l'instruction 'circle' ne soit exécutée qu'une fois sur deux en alternance avec l'instruction 'box'. Pour réaliser cela, vous pouvez dire à Authorware de n'exécuter une instruction que si une condition est vraie. La syntaxe est la suivante:

```
SI <condition> ALORS <instruction>
```

En anglais les termes 'SI... ALORS...' se traduisent par 'IF ... THEN'. Par exemple, la ligne ci-dessous déclenche un beep si le pourcentage de réponses correctes est inférieur à 20.

```
If PercentCorrect < 20 then beep ()
```

Si la variable PercentCorrect n'est pas inférieure à 20, l'instruction 'beep' ne sera simplement pas lue par Authorware. Une forme de condition un peu plus complexe permet de choisir entre deux instructions:

```
SI <condition > ALORS <instruction> SINON <instruction>
```

L'exemple ci-dessus comptabilise les réponses, en ajoutant 3 points au score de l'utilisateur qui répond en moins de 5 secondes et seulement un point s'il répond plus lentement. Notez la présence du 'end if' qui mar-

que la fin de l'expression, de la même manière que le 'end repeat' marque la structure d'une boucle.

```
if ResponseTime < 10 then
   score := score + 3
   else
   score := score + 1
end if
```

Exercice 6-9: Dessin d'une diagonale de carrés

Reprenez le programme 6.8 et sauvez le sous 6.9. Modifiez la boucle de telle sorte que les éléments apparaissant sur la diagonale ne soient pas des disques noir mais des carrés blancs.

Attention, à l'examen, un étudiant sur trois commet l'erreur suivante: elle consiste à confondre une condition qui s'exprime par 'N = 2' et une affectation qui s'exprime par 'N := 2'. Il est vrai que les syntaxes sont fort similaires, mais le sens est complètement différent: **une condition est une question**. ('Est-ce que N est égal à 2?'), alors que **l'affectation est un ordre** ('Donnez à N la valeur 2'). En informatique, au lieu de question, on utilise le terme prédicat. Il désigne toute expression que le système peut évaluer, c'est-à-dire déterminer si cette expression est vraie ou fausse.

Les conditions plus complexes peuvent se construire par la conjonction (&), la disjonction (|) ou la négation (~) d'autre conditions.

```
N  >= 10 & N <=   20 N est-il compris entre 10 et 20?
N < 10 | N > 20      N est-il en dehors de l'intervalle 10-20?
~ (N = 5)            N n'est pas égal à 5?
N  <> 5              N est-il différent de 5?
(N  <= 10 & N  >= 20)|~(X>0)Est-ce que soit N est compris entre 10 et 20, soit X n'est pas
                     supérieur à 0?
```

Pour choisir entre plus de deux instructions, vous pouvez imbriquer des structures IF-THEN les unes dans les autres. Dans l'exemple ci-dessous la condition principale discrimine les scores supérieurs ou inférieurs à 50, ensuite chacune des autres conditions scinde à son tour l'intervalle considéré en deux parties. La figure 6.2 présente l'arbre de décision qui correspond à ces tests imbriqués.

```
if PercentCorrect < 50 then
      if PercentCorrect < 20 then diagnostique := "Désespéré"
         else
         diagnostique := "Grave"
      end if
   else
      if PercentCorrect < 80 then diagnostique :=  "Bon"
         else
         diagnostique := "Parfait"
      end if
end if
```

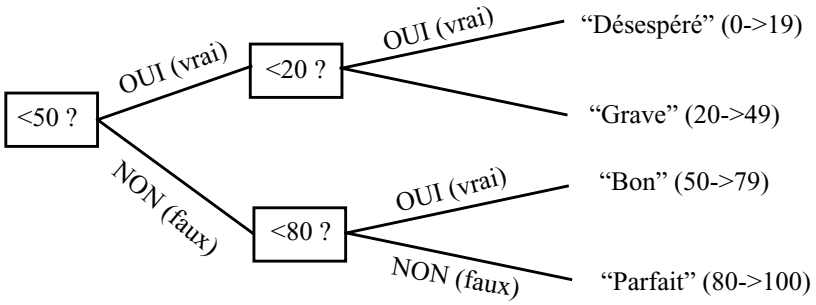


Figure 6-2:Conditions imbriquées

6-8 Algorithmique

En étudiant les structures itératives et conditionnelles, vous venez de découvrir les bases de l'algorithmique. En combinant ces structures de diverses manières, vous pouvez créer une immense variété de séquences d'instructions. Par exemple, une condition permet de définir un critère de sortie prématurée d'une boucle grâce à l'instruction 'exit repeat'[48]. Testez l'effet de la boucle ci-dessous, dans laquelle la boucle sur N2 est interrompue chaque fois que N1 = N2.

```
repeat with n1 := 1 to 20
      repeat with n2 := 1 to 20
          if n1=  n2 then exit repeat
          X := n1 * 20
          Y := n2 * 20
          Circle (-1, X, Y, X + 10, Y +10)
      end repeat
end repeat
```

Exercice 6-10: Message d'attente (suite)

Reprenez le programme 6.7 et sauvez les sous 6.10. Modifiez le message 'Patientez 0.3 secondes'. Interrompez le dessin après exactement 0.3 secondes.

Ces quelques instructions vous permettent donc d'écrire du code de programmation classique dans un programme Authorware. Authorware reste cependant très éloigné de la puissance des langages de programmation. Vous ne pouvez pas définir de procédures paramétrées retournant un produit, pas de procédures récursives, vous ne disposez pas de classes d'objets. Ajoutons encore que la syntaxe décrite ci-dessus n'est pas des plus élégantes et que le 'debugger' d'Authorware ne rend pas compte des itérations successives. Bref, Authorware ne vous offre ici qu'un sous-ensemble des fonctionnalités attendues d'un langage de programmation, ce qui limite sans conteste la puissance de l'outil par rapport à de véritables langages.

Les structures conditionnelles et itératives décrites ci-dessous sont aussi disponibles en programmation graphique, grâce à l'icône de décision étudiée dans la leçon 8.

Pour terminer cette leçon, voici quelques informations qui vous seront utiles:

- Si dans une boucle REPEAT vous modifiez la valeur d'une variable affichée à l'écran, vous ne verrez pas l'évolution de la valeur de cette variable, même si vous avez pertinemment choisi l'option 'Update displayed variables permanently'. La valeur affichée sera réactualisée uniquement à la sortie de la boucle.
- Lorsque vous fermez une icône de calcul, Authorware vous demande systématiquement si vous voulez garder les changements effectués. Cette vérification systématique s'avère rapidement insupportable. Pour l'éviter, ne fermez pas la fenêtre au moyen de la souris, mais utilisez la touche ENTER (généralement placée sur le pavé numérique).
- Dans une application réelle, il est souvent nécessaire d'utiliser de nombreuses icônes de calcul, ce qui rend le programme moins lisible. Une alternative consiste à associer du code à n'importe quelle icône. Sélectionnez une icône, puis sélectionnez l'option '**calculations**' dans le menu 'Modify/Icon'. Vous pouvez alors entrer votre code comme dans une icône de calcul. Lorsque vous fermez la fenêtre, Authorware place simplement une marque '=' à côté de l'icône choisie pour vous rappeler que des instructions y ont été associées. Cette possibilité est particulièrement utile si vous désirez associer un calcul à chaque itération dans une boucle d'interaction. Il n'est en effet pas possible de placer une icône de calcul entre la flèche de retour issue des réponses 'try again' et l'icône d'interaction proprement dite. Vous pouvez par contre associer du code à l'icône d'interaction même, grâce à 'calculations'. Celui-ci sera effectué chaque fois que le sujet fournira une nouvelle réponse.

48. Ne confondez pas 'exit repeat' avec 'exit', laquelle permet de quitter l'icône de calcul sans exécuter les instructions placées en dessous d'elle, ce qui permet de réduire le nombre de conditions à exprimer.

6-9 Commentaires

Nous vous recommandons vivement de laisser des commentaires dans vos icônes de calcul. Un commentaire est un message placé à droite ou en dessous des instructions et qui vous aidera à vous souvenir de l'effet d'une fonction ou du rôle d'une variable. Ne notez pas ces commentaires sur un bout de papier que vous perdrez de toute façon, notez-les à côté de l'instruction concernée, simplement en les séparant par "--". Authorware sait que ce qui se trouve à droite de "--" doit être ignoré pendant l'exécution du programme.

Leçon 7. Intégration des composantes multimédia

7-1 Objectif

Nous vous proposons un moment de détente. Après la leçon difficile sur les variables, voici une leçon facile qui vous permettra de découvrir comment créer des animations et comment intégrer des sons ou des films dans Authorware. Vous ferez connaissance avec quatre nouvelles icônes: animation, film, vidéo et son.

Nous utiliserons occasionnellement les variables dans cette leçon, mais c'est surtout dans la leçon 8 que nous reprendrons l'utilisation intensive de ce que vous avez appris dans leçon 6.

7-2 Philosophie d'Authorware en matière de multimédia

Authorware a été conçu comme un outil qui permet d'**assembler** des composantes multimédia dans un scénario interactif, mais non de développer ces composantes. Celles-ci doivent être réalisées au moyen de logiciels spécialisés permettant respectivement d'éditer des sons, de digitaliser des images vidéo, de créer des animations en trois dimensions, de traiter des images, etc. Les icônes d'Authorware permettent d'insérer facilement ces éléments dans un programme et de régler quelques paramètres déterminant leur exécution.

Ce principe apparaissait déjà dans la toute première leçon: la boîte à outil graphique associée à l'icône de présentation ne permet pas de réaliser du graphisme sophistiqué: pas de rotation d'objets, pas de tracé main libre, pas de dégradés de couleur, etc... Toute mise en page plus sophistiquée doit donc être réalisée dans un logiciel de dessin (SuperPaint, ClarisDraw, CorelDraw, ...) ou de traitement de l'image (Photohop, PixelPaint, ...) et ensuite être importée dans Authorware par copier-coller. Cette approche s'applique également aux aspects multimédias.

Un des paramètres communs aux différentes composantes multimédia concerne la **synchronisation** des icônes. Par défaut, Authorware exécute les icônes les unes après les autres, une icône n'étant lue que lorsque l'icône précédente a été complètement exécutée. Ce mode correspond à l'option **'wait until done'** affichée par défaut lorsque vous éditez une icône son, animation ou film. Ce mode ne permet pas d'exécuter simultanément deux icônes, pas exemple une icône son et une icône film, afin d'obtenir des effets sonores sur le film. Vous pouvez modifier cela en sélectionnant l'option **'concurrent'**. Si une icône est en mode 'concurrent', Authorware lance son exécution mais n'attend pas la fin, il commence immédiatement à exécuter l'icône suivante, le contenu des deux icônes étant donc - dans la mesure du possible exécuté de façon synchrone[49]. L'option **'perpetual'** permet de répéter indéfiniment l'exécution de l'icône (jusqu'au moment où un autre événement l'interrompra).

7-3 L'icône son

Avant de lire ce qui suit, assurez-vous que la machine sur laquelle vous travaillez possède une carte son. Sans cette carte, vous ne pourrez observer les effets de votre programme.

Prenez une icône son , l'avant dernière de la palette, et posez la sur la barre de déroulement.

Double-cliquez sur cette icône. Authorware ouvre une boîte de dialogue permettant de régler certains paramètres.

- En premier lieu, chargez un fichier son en cliquant sur **'load'**. Vous trouverez quelques fichiers disponibles sous ftp://tecfa.unige.ch/pub/sounds/. Une fois le fichier chargé, vous pouvez l'**écouter** en

49. Un léger décalage peut néanmoins se produire si Authorware perd une fraction de seconde pour charger la deuxième icône, par exemple un film important. Vous pouvez atténuer cet effet en chargeant préalablement le contenu de cette deuxième icône grâce à la fonction **'preload'** (à placer dans une icône de calcul).

pressant sur le bouton 'play' (flèche à droite) et l'arrêter en pressant le bouton 'stop' (carré noir). D'autre part, vous obtiendrez toute information utile sur ce son en cliquant sur le bouton 'info' (longueur du son, taux de compression, nombre de bits par mesure, taux d'échantillonage, etc.)

- Vous pouvez modifier la vitesse de ce son (champs 'speed'), 100% étant la vitesse du son dans le fichier chargé. Attention, changer la vitesse déforme le son. Utilisez cette option pour créer des effets sonores originaux.
- Combien de fois Authorware doit-il jouer ce son? Par défaut, il l'exécutera une seule fois (option 'once'). Pour le jouer plusieurs fois, introduisez le nombre de répétitions dans le champs 'times'. Vous pouvez soit introduire un nombre, soit une variable numérique. Par exemple, si vous introduisez la variable 'score' dans le champs 'times', et que cette variable vaut 5, Authorware jouera 5 fois le son. Enfin, le son peut être joué et rejoué jusqu'au moment où une condition est vraie (champs 'until true'). Imaginons un jeu dans lequel un engin se déplace en faisant du bruit et en consommant du carburant. La quantité de carburant est stockée dans la variable 'StockCarburant'. Pour interrompre le bruit du moteur quand le réservoir est vide, introduisez dans le champs 'until true' la condition 'StockCarburant = 0'.
- Vous pouvez comprimer les sons : le bouton 'compress' vous propose deux taux de compressions 'trois fois ' et 'six fois'. Un son compressé prend moins de place dans le système mais sa qualité baisse. Pour évaluer le gain de mémoire à la perte de qualité, comprimez le son, comparez la taille (bouton 'info') et comparez la qualité de son en écoutant les deux versions.
- Enfin, le bouton portant un petit disque rouge vous permet d'enregistrer un son, ce qui est assez pratique pour introduire des consignes orales.[50]

Exercice 7-1: Message vocal

Posez la question oralement "Pressez le bouton CONTINUE dès que vous êtes prêts à commencer". Il faut que le sujet puisse cliquer sur CONTINUE en même temps qu'il entend le message.

Vous notez au passage que la qualité des messages oraux enregistrés par vous ou moi est largement insuffisante. Lorsque vous développez une application professionnelle, vous devrez faire appel pour la lecture des messages à des professionnels de la diction[51].

7-4 L'icône vidéo

Authorware comporte trois icônes permettant de réaliser des animations avec plus ou moins de complexité. La première, l'icône vidéo, vous permet de contrôler un magnétoscope externe. La machine-cible pour laquelle vous développez une application doit posséder une interface spéciale (hardware) vers le magnétoscope. L'édition de l'icône vidéo vous permet de préciser quelle partie de la bande doit être lue (numéros de la première et dernière 'frame'), de contrôler la vitesse, de choisir les canaux sonores, etc. Vous pouvez aussi demander à Authorware d'afficher à l'utilisateur une barre de commandes présentée ci-dessous, comprenant les boutons classiques d'un magnétoscope.

Nous n'insistons pas davantage sur cette icône car elle est de moins en moins souvent utilisée. D'une part, la connexion entre l'ordinateur et le magnétoscope est un problème technique pénible (il faut savoir quelle carte pour quel ordinateur convient pour quel magnétoscope). D'autre part, les séquences animées sont de

50. A propos de Design : Présenter oralement les instructions permet à l'utilisateur de maintenir son attention visuelle sur les éléments de l'interface cités dans les instructions. Par contre, il s'agit d'une information volatile, alors qu'une instruction présentée visuellement peut être relue de multiples fois. Aussi, si vous avez recours à des instructions orales, prévoyez un bouton permettant à l'utilisateur d'écouter à nouveau le message oral.

51. Enregistrez vous-même les messages pour développer le logiciel. Faites seulement appel à un professionnel pour enregistrer ces messages lorsque l'application est au stade ultime de son développement.

plus en plus souvent digitalisées et stockées sur CD-ROM. Dans ce cas, elles seront insérées dans votre logiciel au moyen de l'icône 'film' que nous étudions ci-après.

7-5 L'icône film

L'icône 'film' fonctionne de façon très semblable à l'icône son: vous chargez un film créé à l'extérieur d'Authorware, vous le collez dans Auhorware et ensuite vous en réglez les paramètres. Le film peut être une séquence vidéo digitalisée ou une animation créée avec un autre logiciel. Authorware intègre particulièrement bien les animations crées au moyen de DIRECTOR, car il s'agit d'un logiciel produit par la même société (Macromedia). Authorware supporte en outre tout film, quelle qu'en soit l'origine, pour autant qu'il soit réalisé dans un des formats suivants: sur Mac, QuickTime, Pics, Macintosh Movie Editor (.mve), sous Windows, AVI, QuickTime for Windows (.mov), Autodesk Animator (.flc .fli et .cel).

L'intégration du film dans le logiciel dépend de différents paramètres que vous pouvez régler en éditant l'icône film:

- Tout film ou animation se compose d'une succession de **'frames'**, chaque frame étant un dessin ou une image. Vous pouvez jouer partiellement le film en précisant le numéro de la première et de la dernière frame ou jouer ce film pas-à-pas au moyen du bouton 'step'.
- La vitesse d'un film est le nombre de frames présentées par seconde. Le standard cinématographique, est de 24 images par secondes. Si vous réduisez le nombre d'images, l'utilisateur ne percevra peut-être plus la continuité des images, mais une succession d'images saccadées. Si l'ordinateur n'arrive pas à afficher les images au rythme prévu, il 'saute' certaines images pour maintenir le rythme demandé, ce qui peut également produire un effet de saccade. Pour éviter cela, sélectionnez l'option 'don't skip frames' (Celle-ci peut ralentir l'exécution du film).
- Par défaut, Authorware joue le film en boucle, revenant à la première frame dès qu'il a joué la dernière (option 'repeatedly'). Vous pouvez régler différemment le nombre d'exécutions: jouer n fois ('? times') ou, comme pour le son, jusqu'au moment où une condition est vraie. L'option times/cycle permet de contrôler le nombre de fois qu'un film est montré sur la durée d'une animation. Par exemple, un film qui représente la rotation de la lune est montré dix fois pendant l'animation de d'une révolution de la lune autour de la terre. L'option 'only while in motion' sera décrite ultérieurement. Enfin, vous pouvez donner le contrôle du déroulement du film à l'utilisateur (option 'under user control'). Authorware affiche alors une barre de contrôle pendant l'exécution. Celle-ci permet de jouer un film en avant en arrière ou de faire un arrêt sur image.
- Si le film comprend lui-même des élément interactifs que vous désirez rendre accessible à l'utilisateur, sélectionnez l'option **'Interactivity: ON'**. Cette option s'applique uniquement aux films créés au moyen de Director, un outil de développement vendu par la même société qu'Authorware.
- Vous pouvez encore déterminer si le film apparaît en transparence ou en mode opaque, et si le son éventuel doit être joué. Enfin, vous disposez des options de synchronisation ('timing') identiques pour toutes les icônes multimédia.

Exercice 7-2: Synchronisation film-son

Choisissez un film et un son puis synchronisez leur présentation. Vous trouverez des films et des sons à l'adresse suivante: ftp://tecfa.unige.ch/pub/mpeg et ftp://tecfa.unige.ch/pub/sounds

7-6 Animations à destination fixe

Vous pouvez réaliser des animations au sein même d'Authorware grâce à l'icône d'animation . Celle-ci permet de déplacer 'à plat' le contenu d'une icône de présentation, selon différentes modalités. Elle ne permet pas de faire des rotations ou autres effets, ni en deux , ni en trois dimensions. Je vous propose de construire ensemble votre première animation.

Exercice 7-3: Animation à destination fixe

Nous allons créer une animation dans laquelle un disque rouge traverse l'écran.

Premier pas. Dessinez l'objet à déplacer. Comme pour les réponses par déplacement d'objet, l'objet à animer doit figurer seul dans une icône de présentation. Rappelez-vous qu'Authorware fonctionne en mode calque: tous les objets appartenant à une même icône seront animés solidairement, comme s'ils ne constituaient qu'un seul objet. Créez un disque plein sur une icône de présentation. J'insiste sur le fait que le disque soit plein car cela facilite les manipulations ultérieures. Appelez cette icône 'disque'.

Second pas. Positionnez une icône d'animation après l'icône 'disque'. Double-cliquez sur cette icône. Authorware ouvre une fenêtre en haut de laquelle vous avez les message 'drag object to destination' ('déplacez l'objet à destination'). Cliquez sur le disque rouge et déplacez-le de l'autre côté de l'écran. Si le disque rouge n'apparaît pas, ouvrez à nouveau l'icône 'disque' et recommencez ce deuxième pas.

Troisième pas. Cliquez sur 'replay' et Authorware reproduit le déplacement que vous avez effectué, en prenant la ligne droite entre le point de départ et le point d'arrivée. La durée choisir pour le déplacement est celle que vous avez mis pour démontrer le trajet.

Quatrième pas. Exécutez votre programme. Le disque rouge traverse l'écran selon vos spécifications.

Authorware vous permet de régler la vitesse du déplacement ('rate') de deux façons différentes: soit en fixant la vitesse même de l'objet, vitesse exprimée en pixel par secondes (option 'speed'), soit en déterminant la durée totale du déplacement (option 'time'). A ce stade, ces deux options sont équivalentes, mais nous verrons ultérieurement que ce n'est pas toujours le cas.

L'icône d'animation vous permet de régler le mode de synchronisation (‘wait until done’ / ‘concurrent’) comme pour les icônes précédentes. Elles vous permet aussi d'attribuer un 'layer' à chaque animation: dans l'exemple de la figure 7-1, le disque blanc animé au layer 2 passera devant le rectangle gris animé au layer 1. L'effet présenté dans cette figure repose sur des objets en mode 'opaque', vous obtiendrez bien sûr des effets de superposition différents en mode transparent ou dans d'autres modes. Vous pouvez utiliser plus de deux plans ('layers').

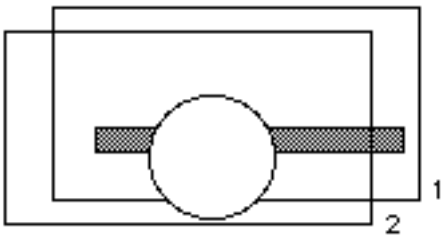


Figure 7-1:Arrière-plan et avant-plan ('layers') dans l'animation

Dans l'animation que vous venez de créer, Authorware déplace l'objet en ligne droite depuis son point de départ jusqu'à son point d'arrivée. Il s'agit du type d'animation 'direct to fixed point'. Authorware vous offre en fait quatre autres types d'animations. Vous pouvez par exemple faire déplacer un objet entre deux points mais selon une **trajectoire précise** que vous dessinez. Quand vous éditez l'icône d'animation, cliquez sur 'change type' et sélectionner l'option 'along path to fixed point'.

Une animation de type 'trajectoire' possède les même paramètres que le type précédent, mais vous devez éditer la trajectoire. Pour cela, déplacez l'objet et chaque fois que vous le laissez tomber, Authorware dépose une marque triangulaire. En mode édition, la trajectoire est visible et modifiable, elle n'est par contre pas visible pour l'utilisateur en mode exécution. Reprenez l'objet, glissez-le le long de la trajectoire pour le déposer plus loin, etc[52]. Un déplacement peut comprendre un grand nombre de marques triangulaires. Par défaut, Authorware déplace les objets en ligne droite entre deux postes intermédiaires. Si vous cliquez deux

fois sur un poste intermédiaire, les segments de trajectoire associés à ce poste seront courbes. Un poste courbe est représenté par un petit cercle. Cliquez à nouveau pour revenir à des segments droits. Vous pouvez ajouter un poste en cliquant sur la trajectoire, retirer un poste ('delete point') ou le déplacer.

Exercice 7-4: Labyrinthe

Dessinez un labyrinthe. Ensuite dessinez un animal qui s'y déplace et se heurte à différentes voies sans issue.

Ce type d'animation dispose d'une option de synchronisation supplémentaire: 'perpetual'. Dans ce cas, l'objet recommence inlassablement la même trajectoire. Il est alors utile d'utiliser l'option 'move when true': placez une condition et, lorsque celle-ci ne sera plus vérifiée, l'animation s'arrêtera.

Exercice 7-5: Cible mouvante

Créez une cible qui se déplace sur l'écran de façon continue. Au moment où le sujet clique dessus, l'utilisateur entend un grand 'boum'.

Vous constatez dans l'exercice 7-5 que l'animation ne s'arrête pas immédiatement, l'objet terminant en fait la trajectoire spécifiée. En fait, il n'est pas possible de vérifier en permanence si la condition est vraie. Celle-ci est vérifiée à chaque fois que l'objet recommence la trajectoire. Une solution élégante consisterait à effacer l'objet au moment où l'utilisateur clique dessus.

7-7 Animations à destination variable

Dans les deux types d'animation que vous avez utilisées, l'objet se déplace systématiquement jusqu'au bout de sa trajectoire. Les deux types d'animation suivants, 'to calculated point on line' et 'to calculated point on path', vous permettent d'arrêter l'objet sur une position intermédiaire en fonction de la valeur d'une variable. Par exemple, si le point initial de la trajectoire vaut 0 et le point final vaut 100, et si la variable de positionnement vaut 66, l'objet s'arrêtera après avoir parcouru deux tiers du chemin.

Pour permettre à Authorware d'interpoler la position de l'objet entre les deux extrêmes, vous devez fournir trois valeurs numériques: le nombre correspondant au point de départ ('base'), au point d'arrivée ('end') et enfin la variable dont la valeur sera utilisée pour calculer la position intermédiaire. Pour éditer la trajectoire, cliquez d'abord sur le bouton-radio 'base', déplacez l'objet à son point de départ, ensuite cliquez sur le bouton-radio 'end' et déplacez l'objet à son point terminal. Par exemple, pour représenter une horloge sur laquelle un petit point se déplace, vous créez une trajectoire circulaire dont la base vaut 0 et la fin 12 et dont la position est déterminée par la variable-système 'hour'.

Exercice 7-6: Véhicule sur dents de scie

Créez une animation sur une trajectoire en dent de scie sur laquelle se déplace un véhicule quelconque. Deux boutons sont présentés à l'utilisateur: 'avancer vers la droite' et 'avancer vers la gauche'. L'objet se déplace dans la direction choisie sur une distance équivalente à 10% de la trajectoire totale. Ensuite, l'utilisateur peut presser à nouveau un bouton et ainsi de suite. Désactivez le bouton 'vers la droite' lorsque le véhicule atteint l'extrémité droite de la trajectoire, faites de même pour le bouton et l'extrémité de gauche.

Dans l'exercice 7-6, en désactivant les boutons, vous empêchez la variable 'position' de prendre une valeur en dehors de l'intervalle 0-10. Ce n'est pas toujours possible. Que se passe-t-il si la variable prend une valeur de dehors de l'intervalle numérique défini? Pour reprendre l'exemple ci-dessus, si la variable 'hour' vaut 15, le curseur se déplacera jusqu'à la position 12 et s'arrêtera donc au bout de la trajectoire. Par contre, si vous sélectionnez l'option **'loop'**, il s'arrêtera à la position 3. Le curseur ne fait pas tout le tour de l'horloge, mais

52. Si vous déplacez l'objet en marche arrière, Authorware ne vous laissera pas sortir l'objet de la trajectoire représentée à l'écran. Pour retrouver la liberté de mouvement, sortez l'objet de sa trajectoire pas le dernier poste créé.

simplement un déplacement identique à celui qu'il aurait effectué si 'hour' eut valu 3. Il se serait également arrêté à 3 pour les valeurs 27, 39,... Bref, il remplace la valeur N par N 'modulo 12'. Dans le cas d'une trajectoire en ligne droite, vous disposez des mêmes options 'loop' et '**stop at ends**', et d'une option supplémentaire: '**go past ends**'. Dans ce dernier cas, si la valeur sort de l'intervalle fixé, Authorware extrapole la position de l'objet en respectant l'axe de la trajectoire défini par l'auteur.

Exercice 7-7: Curseur et cercle

Dessinez un cercle sur lequel l'utilisateur déplace un curseur au moyen des boutons '<', '<<', '>', '>>'. Ces boutons déplacent le curseur de 10% et 30% de la trajectoire, soit dans le sens des aiguilles d'une montre, soit dans le sens opposé. Lorsque le curseur aboutit à un extrémité de la boucle, il continue au-delà de celle-ci. Réglez la vitesse sur 'time: 1 second'.

En réalisant l'exercice 7-7, vous avez constaté que lorsque l'objet est positionné par exemple au 95% de sa trajectoire et doit se rendre au point 5% (ou - ce qui est équivalent au 105% en mode 'loop'), il revient en arrière sur la trajectoire. Vous ne pouvez modifier cela.

Toujours à propos de l'exercice 7-7, vous avez probablement aussi constaté que la vitesse de l'objet n'était pas la même lorsque vous pressiez sur '>' ou '>>'. En effet, il doit parcourir 10% de la trajectoire dans le premier cas et 30% dans le second, alors que la durée est la même. A durée égale, le déplacement sera d'autant plus rapide que la distance est longue. Si vous voulez assurer que les déplacements se fassent tous à le même vitesse, définissez une vitesse absolue au moyen de l'option '**speed**'. Inversement, si vous désirez garantir que tous les déplacements se produisent dans la même durée, fixez une durée absolue (option '**time**').

7-8 Animations dans le plan

Le dernier type d'animation consiste à se rendre en ligne droite à un point de l'écran ('direct to grid') mais dont la position est calculée par rapport à une grille dont vous établissez les dimensions. Procédez comme précédemment: cliquez sur le bouton 'base' et positionnez l'objet dans ce qui constituera l'origine des axes. Ensuite, cliquez sur le bouton 'end' et déplacez l'objet vers le point correspondant à l'extrémité des deux axes. La ligne séparant les positions 'base' et 'end' constituent la diagonale de la grille qu'Authorware utilisera pour positionner les objets. Il s'agit d'une grille fictive: l'auteur en perçoit le cadre, l'utilisateur ne la perçoit pas. Ensuite, vous devez définir la valeur inférieure, la valeur supérieure et la variable de positionnement, comme c'était le cas précédemment, mais vous devez le faire deux fois, une pour l'axe horizontal et une pour l'axe vertical.

Exercice 7-8: Agenda

Dessinez un agenda de la semaine divisé en sept colonnes (une par jour) et 8 rangées (8 h, 9h, 10 h, 11 ., 14 h, 15 h, 16 h et 17 h. Créer une marque qui se déplace vers la zone correspondant au jour et à l'heure réelle. Présentez quatre boutons pour déplacer la marque: 'jour suivant', jour précédent, 'heure suivante' et 'heure précédente'.

7-9 Positionnement variable sans animation

Vous avez pu constater dans l'exercice 7.6 que l'objet animé apparaît d'abord toujours à l'endroit où il a été dessiné avant de se positionner dans la grille en fonction des variables déterminant sa position. Authorware vous permet de fixer aussi la position initiale d'un objet au moyen de variables. Lorsque vous créez l'objet dans une icône de présentation, sélectionnez l'objet puis cliquez sur l'option '**icon/properties**' du menu 'Modify'. Dans cette boîte de dialogue, l'option '**positioning**' est par défaut sur la position 'no change'. Cela signifie que l'objet apparaît exactement à l'endroit où l'auteur l'a positionné. L'option 'on path' vous permet de faire apparaître l'objet à un endroit variable sur un axe. L'option 'in area' fait de même sur deux axes. Vous spécifiez le ou les axes et la position de l'objet exactement de la même façon que dans l'icône d'animation. La seule différence est que l'objet ne glisse pas vers la position calculée, il apparaît directement à cette po-

sition.

Exercice 7-9: Agenda (suite)

Modifiez l'exercice 7.8 le sauvez-le sous 7.9. de telle sorte que la marque apparaisse initialement (sans animation) au jour et à l'heure correspondant au jour et à l'heure réels.

7-10Déplacer un film

L'icône d'animation permet de déplacer un film. Par exemple, le film 'AirplaneTalingOff.QT' montre un avion qui décole tout en restant à la même position horizontale. Vous pouvez sélectionner ce film dans une icône d'animation, afin de donner l'impression que l'avion avance.

Cette combinaison pose parfois des problèmes de synchronisation, par exemple, un cheval au galop continue à courir alors que le déplacement est arrivé à son terme, donnant l'impression de courir sur place. L'option 'Play movie: only while in motion' permet d'interrompre l'excécution du film lorsque le déplacement s'arrête.

7-11Librairies

Vous le constaterez rapidement, insérer un son ou une image augmente rapidement la taille de votre programme. Or, il arrive que le même son, le même film ou la même image soient utilisés 25 fois dans le même programme. Si une icône est appelée à être utilisée plusieurs fois, placez l'icône originale dans une librairie externe et placez dans votre programme un lien vers cette librairie. Je vous propose de créer ensemble votre première librairie.

Exercice 7-10: Création d'une Librairie

Premier pas. Créez une icône son et placez-la dans votre programme. Sélectionnez un son quelconque. Appelez-la 'Son1'

Deuxième pas. Sélectionnez 'new/library' dans le menu 'File'. Authorware crée une nouvelle fenêtre.

Troisième pas. Glissez l'icône 'Son1' du programme vers la librairie. Vous remarquez que le nom de l'icône dans le programme apparaît maintenant en italique. Cela signifie que l'original se trouve dans la librairie, mais que sa copie -ou son alias - se trouve dans le programme. A côté de l'original figure le symbole *qui indique un 'lien' ('link') entre l'icône originale et son utilisation dans le programme. Sous le nom de l'icône dans la librairie, vous pouvez laisser un commentaire. Il est utile de décrire en quelques mots le contenu de l'icône.*

Quatrième pas. Glissez l'icône 'Son1' qui figure dans la librairie vers votre programme. Authorware place un deuxième copie de 'Son1' dans le programme. Recommencez et Authorware place une troisième copie de 'Son1' dans le programme.

Cinquième pas. Effacez une icône du programme (sélectionnez l'icône et pressez 'delete'). Cela ne pose pas de problème.

Sixième pas. Effacez l'icône 'Son1' qui figure dans la librairie. Vous créez alors le chaos! En effet, en exécutant le programme, Authorware trouvera les liens vers la librairie mais n'y trouvera pas l'original. Ces liens brisés sont représentés dans le programme par un icône .

Seules les icônes suivantes peuvent être stockées dans une librairie: présentation, interaction, son, film et calcul. Les icônes présentation et interaction y seront stockées lorsqu'elles contiennent une image gourmande en information (par exemple, une grande image en millions de couleur).

Une librairie ne permet pas seulement d'économiser de l'espace-mémoire mais également du temps de travail. Imaginons que vous dupliquiez 15 fois une icône film dans votre programme. Ensuite, vous décidez

de modifier la vitesse du film. Vous devez dans ce cas éditer une par une chacune des quinze icônes. Par contre, s'il s'agit d'une icône unique avec quinze liens, il vous suffira de modifier l'icône de la librairie: sélectionnez 'library links' du menu 'Xtras'; sélectionnez les liens ('unbroken links') à rafraîchir et cliquez sur 'update'.

Précisons enfin que l'utilisation d'icônes de calcul dans une librairie permet d'appeler certaines fonctions complexes depuis plusieurs endroits et compense en cela partiellement l'absence de procédures.

Leçon 8. Structuration du logiciel

8-1 Objectif

Après cette récréation multimédia, retour aux structures algorithmiques: l'itération et la condition. Vous avez appris à construire ces structures au moyen de fonctions placées dans une icône de calcul. Dans cette leçon, vous apprendrez à réaliser des structures équivalentes au moyen de l'icône de décision.

Rappel: l'icône sous-programme vous permet de grouper plusieurs icônes.

8-2 Créer une boucle composée d'icônes

L'icône de décision fonctionne un peu comme l'icône d'interaction. Nous vous proposons de réaliser ensemble le premier exercice.

Exercice 8-1: Cible mouvante itérée

Comment recommencer plusieurs fois la même question? Voici un exemple.

Premier pas. Chargez le programme 7.5, dans lequel le sujet devait cliquer sur un objet en déplacement. Regroupez les différentes icônes dans une seule icône 'sous-programme' que vous dénommez 'ClickCible'. Sauvez-le programme sous 8.1.

Deuxième pas. Posez une icône de décision sur la barre de déroulement, peu importe où. Prenez l'icône 'ClickCible' (le sous-programme) et déposez-la à droite de l'icône de décision. Authorware construit un algorithme que vous pourrez ensuite modifier en réglant les différents paramètres de cette icône.

Troisième pas. Double-cliquer sur l'icône de décision. Les paramètres listés dans la zone intitulée 'repeat' vous offrent plusieurs façons de déterminer le nombre d'itérations. L'option par défaut 'Don't repeat' implique un seul passage dans le corps de la boucle. Cliquez sur la première option et introduisez '5' dans le champ juste en dessous. Authorware modifie le dessin de l'algorithme pour représenter la boucle. Faites un run. Le programme répète cinq fois l'interaction conçue dans le programme 7.5.

D'autres critères sont disponibles pour déterminer le nombre de répétitions. Le critère 'until true' consiste à répéter la boucle jusqu'au moment où une condition, exprimée au moyen de variables, est vérifiée.

Exercice 8-2: Cible mouvante itérée (suite)

Modifiez l'exercice 8.1 et sauvez-le sous 8.2. L'utilisateur recommence l'exercice jusqu'au moment où il cumule trois succès.

8-3 Tracer le fonctionnement d'Authorware

Pour réaliser les exercices qui suivent, le **'debugger'** d'Authorware vous sera précieux. Sélectionnez l'option 'trace window' dans le menu 'try it'. En haut de la fenêtre 'trace', juste sous la barre des boutons, est affichée l'icône qu'Authorware est en train d'exécuter. Dans la fenêtre principale, Authorware décrit chaque icône rencontrée:

- Le numéro en tête de ligne est le niveau de l'icône: se trouve-t-elle sur la barre de déroulement du programme principal (niveau 1), ou dans un sous-programme (niveau 2), un sous-sous-programme (niveau 3), ...?

- Le deuxième élément décrit le type d'icône. Les codes utilisés sont listés dans le manuel Authorware.
- Le troisième élément contient le nom de l'icône activée et les actions éventuelles de l'utilisateur.

Si vous ne comprenez pas ce que votre programme fait, essayez de suivre pas-à-pas la trace de son raisonnement. Malheureusement, cette trace ne décrit pas ce qui se passe à l'intérieur d'une icône: vous ne pourrez donc pas suivre les différentes itérations au sein d'une icône de calcul.

8-4 Sélectionner des icônes

Dans les exercices précédents, vous n'avez placé qu'une seule icône à droite de l'icône de décision. Si vous placez plusieurs icônes, vous pouvez régler non seulement le nombre de répétitions mais aussi le critère et l'ordre de sélection des différents icônes/branches. Le mode de sélection par défaut consiste à considérer les branches selon leur position dans l'icône (**'Branch: Sequentially'**): la première branche est choisie au premier passage, la deuxième au deuxième passage, etc.

L'ordre de sélection des branches peut être aléatoire. Pour une sélection aléatoire, choisissez entre **'Branch: random to any path'** ou **'Branch: random to any unused path'**. Dans le premier mode, Authorware choisira éventuellement deux fois la même branche, pas dans le second. Si vous créez 5 branches, sélectionnables en mode aléatoire sans répétition, vous devrez fixer le nombre d'itérations à 5 si vous voulez être sûr que chaque branche soit choisie une et une seule fois. Par contre, si vous optez pour le mode aléatoire avec répétition, vous ne pourrez garantir que toutes les branches soient sélectionnées même en demandant 467 itérations! Si votre souci est de garantir que chaque branche soit sélectionnée au moins une fois, choisissez l'option **'Repeat: untill all paths used'** (recommencer jusqu'au moment où toutes les branches ont été utilisées). Mais attention, si vous avez 2 branches, il est possible - peu probable mais possible - que la première branche soit sélectionnée 28 fois avant que la première soit également sélectionnée! Aléatoire... ça veut dire aléatoire!

Exercice 8-3: Formes aléatoires

Créez un programme qui présente six formes géométriques dans un ordre aléatoire.

Lorsque vous éditez une icône placée dans une structure de décision, Authorware vous demande de fixer deux paramètres[53]: (1) faut-il insérer une pause (bouton 'continue') avant l'itération suivante, (2) faut-il effacer les objets affichés dans cette branche avant l'itération suivante, ou seulement à la sortie ou ne pas les effacer ? Ces options vous sont déjà familières, étant semblables à celles utilisées dans l'icône d'interaction. Nous vous recommandons à nouveau de préférer un effacement automatique géré par Authorware, grâce aux options susmentionnées, plutôt qu'un effacement 'manuel' au moyen d'icônes d'effacement placées ici et là.

Généralement, les icônes d'Authorware fonctionnent avec des variables locales: les variables permettant de gérer une icône sont reinitialisées si le programme quitte cette icône et y revient plus tard dans une boucle. Considérons par exemple une icône d'interaction dans laquelle le nombre d'essais est limité à 3. Si cette interaction est elle-même placée dans une icône de décision, le nombre d'essais est remis à zéro lorsque le programme quitte cette icône (puis y revient). Par contre, l'icône de sélection peut se souvenir des branchements effectués préalablement. Ceci est important lorsque vous choisissez un mode de sélection aléatoire sans répétition pour lequel Authorware doit se souvenir des branches préalablement sélectionnées. Si l'option 'reset path on entry' n'est pas sélectionnée, Authorware ne choisira pas deux fois la même branche, même s'il quitte et revient ultérieurement à l'icône de décision. Si l'option est sélectionnée, Authorware évitera les répétitions tant que le programme boucle au sein de la même icône de décision, mais il oubliera les branches préalablement sélectionnées dès qu'il quittera cette icône.

53. Cette boîte de dialogue devient pénible lorsque vous désirez éditer de multiples branches. Pour l'éviter, enfoncez la touche 'Ctrl ' lorsque vous double-cliquez sur une icône placée sur une branche.

Exercice 8-4: Fonds d'écran aléatoires

Répétez trois fois l'exercice suivant. Primo, le programme choisit un fond d'écran au hasard parmi différents fonds de couleurs différentes. Ensuite, il demande à l'utilisateur de cliquer sur un objet qui traverse l'écran. Si l'utilisateur attrape l'objet, l'objet est effacé et on recommence l'exercice. Répéter 3 fois avant de passer à l'exercice suivant. Eviter que la même couleur de fond soit sélectionnée deux fois.

Imaginons que vous posiez une icône de décision avec 5 branches, en mode de sélection aléatoire sans répétition, mais que vous demandez 10 itérations de la boucle. Dans ce cas, pendant les cinq premières itérations, Authorware choisira chacune des 5 branches en évitant de prendre deux fois la même. Ensuite, pour réaliser malgré tout les 10 itérations, il 'oubliera' les branches préalablement sélectionnées, et re-sélectionnera à nouveau aléatoirement les 5 branches disponibles.

Enfin, le dernier mode consiste à sélectionner une branche en fonction de la valeur d'une variable ('**Branch: to calculated path**'). Si la variable (ou l'expression) vaut 1, Authorware sélectionne la première branche; si elle vaut 2, il prend la seconde branche, etc. Si la variable vaut 0, aucune branche n'est sélectionnée, le programme passant directement à l'icône qui suit l'icône de décision. Ce mode de sélection sera précieux pour créer des logiciels qui s'adaptent au niveau de l'utilisateur, comme dans l'exercice ci-dessous.

Exercice 8-5: Objet cliquable de taille variable

Demandez au sujet de cliquer sur un objet de 5 mm de large. Si le temps de réponse est supérieur à 3 secondes, proposez la même activité avec un objet plus grand. Si le temps de réponse est inférieur à 3 secondes, faites de même avec un objet plus petit.

8-5 Time limits

Vous pouvez enfin définir une limite de temps au-delà de laquelle Authorware cesse le processus de sélection/répétition et passe à l'icône suivante.

8-6 La fonction aléatoire

L'icône de sélection permet de choisir au hasard une icône ou un groupe d'icônes parmi plusieurs. Nous l'avons utilisé par exemple pour choisir un dessin parmi d'autres. Dans une application interactive, il est souvent nécessaire de générer des nombres de manière aléatoire. Pour donner à N une valeur au hasard parmi trois valeurs possibles, vous pouvez utiliser l'icône de décision et y placer trois icônes de calcul, par exemple N:= 13, N := 376 et N:= 8763. Par contre, si N est un nombre à tirer au hasard entre 1 et 1000, vous n'allez pas créer mille icônes N:=1, N:=2, ... jusque N:= 1000! Authorware possède, comme tous les langages de programmation, une fonction permettant de tirer un nombre au hasard:

Random(min, max, unités)	tire un nombre au hasard entre 'min' et 'max', avec un pas de'unités'
N:= random (1, 100, 1)	-> N pourra valoir 1, 66, 87, ..., 100
N:= random (9, 10, 0.1)	-> N pourra valoir 9, 9.1, 9.3, ..., 10
N:= random (0, 100, 5)	-> N pourra valoir 0, 5, 25, 50, 75, ..., 100

Exercice 8-6: Zombie

Créez un 'zombie', c'est-à-dire un objet qui se déplace de façon aléatoire dans l'écran.

Attention, la fonction aléatoire est très pratique, mais elle est... aléatoire. Par exemple si vous vous créez une addition de deux nombres au hasard entre 1 et 100, vous obtiendrez tout aussi bien 1+1 que 87+95. Vous pouvez contrôler partiellement cet effet, par exemple en générant le deuxième nombre en fonction du premier:

N1 := random (1, 10,1)

N2 := random (1, 10-N1, 1) générera un nombre tel que N1 + N2 < 10.

Exercice 8-7: Addition aléatoire

Affichez successivement des additions de deux nombres inférieurs à 100 dont la somme est supérieure à 100. Répéter l'addition 5 fois.

Moins nombreuses seront les valeurs possibles, plus grande sera la probabilité de générer les même valeurs plusieurs fois successivement. Dans l'icône de décision, vous pouvez demander à Authorware de ne pas répéter deux fois la même sélection. Par contre, lorsque vous utilisez la fonction aléatoire, vous pouvez éviter que le même calcul soit présenté deux fois consécutivement en intégrant la génération aléatoire dans une boucle qui se répète si les deux nombres générés sont identiques aux précédents.

repeat while N1=oldN1 & N2=oldN2 recommencer si génère les mêmes nombres

N1 := Random (1,2,1)

N2 := Random (1,2,1)

Beep()

end repeat

oldN1 := N1 archive la valeur de N1 et N2 pour la

oldN2 := N2 prochaine génération

La fonction ci-dessus inclut un 'bip' qui vous renseigne sur le nombre d'itérations que le programme a dû effectuer pour générer deux nombres différents des précédents. Cette fonction produit un calcul différent du précédent mais n'empêche pas que le calcul généré soit identique à l'avant-dernier. Dans ce cas, il y a une solution plus complexe qui consiste à enregistrer chaque calcul dans un vecteur de variables (Cf. leçon 12.)

8-7 Les fonctions 'caractères'

Jusqu'à présent, nous avons utilisé les variables uniquement pour traiter des nombres. En réalité elles s'appliquent également au traitement des chaînes de caractères. Le terme 'chaîne de caractères' est utilisé en informatique pour désigner un groupe de caractères quelconques (lettres, caractères spéciaux, espaces, ...). Un mot, une phrase et un texte sont des chaînes de caractères. Voici quelques fonctions permettant de manipuler des chaînes.

Find (chaîne1, chaîne2)	donne 0 si chaîne1 ne se trouve pas dans chaîne2, sinon donne la position de chaîne1 dans chaîne2. Exemple find ("t","rateau" donne 3. 'chaîne1' peut être un pattern incluant des ? et * comme dans l'analyse de réponse-texte.
chaîne1^chaîne2	concatène les deux chaînes en une seule, par exemple "Che"^"min" produit "Chemin"
Substr(chaîne,N1,N2)	donne la partie de la chaîne comprise entre les positions N1 et N2. Par exemple Substr ("chemin", 2,3) produit "he".
Char (N)	donne la lettre dont le code ASCII est N. Le code ASCII de a est 97 et

	de z est 122. Par exemple, char(98) donne "b".
CharCount(chaîne)	donne le nombre de caractères dans une chaîne. Par exemple, CharCount ("un chemin") donne 9.

Exercice 8-8: Mot aléatoire

Construire un mot de 5 lettres différentes choisies aléatoirement. Répéter l'action 10 fois, puis sortir.

8-8 Icônes ou fonctions?

Résumons! Authorware offre deux manières de créer des boucles: avec la fonction REPEAT dans une icône de calcul, ou avec les différents paramètres de REPEAT dans l'icône de décision. De même, il existe deux manières de faire choisir Authorware entre deux ou plusieurs actions: avec la fonction IF-THEN-ELSE dans une icône de calcul ou avec l'option 'Branch: to calculated path' dans une icône de décision. Ces structures de programmation sont donc disponibles sous forme de code de programmation classique (icône de calcul) ou sous forme de programmation graphique (icône de décision). Quel mode choisir ? Tout dépend de éléments manipulés dans cette structure. Une boucle avec la fonction 'calcul' est pertinente si les éléments a répéter s'expriment aisément par d'autres fonctions Authorware, telles que les dessins simples (box, circle, ...) utilisés dans le chapitre 6. Inversement, la solution 'icône de décision' est préférable lorsque les éléments à répéter sont réalisés au moyen des icônes d'Authorware (par exemple, un dessin complexe, une interaction, une animation, ...).

Le même principe s'applique au processus de sélection aléatoire. L'icône de sélection permet de choisir au hasard une icône ou un groupe d'icônes parmi plusieurs icônes. La fonction 'random' permet de générer des valeurs numériques. L'icône de sélection sera préférée dans les cas suivants:

- lorsque le nombre de choix possibles est restreint;
- lorsque le matériel à sélectionner ne peut être construit à partir des valeurs numériques, par exemple, lorsque chaque item sélectionnable est une séquence interactive;
- lorsque le matériel est sujet à des contraintes complexes, par exemple lorsqu'il faut générer quatre nombres tels qu'ils permettent de dessiner un rectangle dont les côtés ne sont ni proche de l'horizontale, ni proche de la verticale:
- lorsque vous voulez exercer un contrôle précis sur les données sélectionnées, tel qu'éviter les répétitions, produire des exercices de difficulté croissante, ...

8-9 Les modèles

Lorsque vous prévoyez 10 zones de réponse sur un graphique, ou lorsque vous créez 10 stimuli semblables dans une icône de décision, il est tentant de réaliser les 9 icônes en copiant puis en collant la première. Est-il préférable de dupliquer une icône de la modifier manuellement ou d'en générer les éléments au moyen de variables et de fonctions? Cela dépend du nombre de différences entre deux icônes. Si les icônes diffèrent par un élément de texte ou la dimension d'un rectangle, créez une icône unique dont les éléments sont des variables ou des fonctions. Si par contre, elles diffèrent par de multiples détails, il est préférable de les dupliquer et de les éditer manuellement.

Toutefois, si vous choisissez copier-coller, faites attention à ce qui suit. Imaginons que vous dupliquiez la première réponse d'une icône d'interaction, puis que vous modifiez le texte de chaque feed-back. Admettons que par la suite, vous désiriez modifier la position ou la taille des caractères des feed-back, vous devrez éditer une par une chacune des icônes dupliquées. Aussi, testez toujours une icône avant de la dupliquer.

Vous pouvez également faire du copier-coller depuis un autre programme Authorware. Cela pose toutefois certains problèmes, notamment lorsque certaines variables des icônes copiées portent le même nom que celles du nouveau programme. Dans ce cas travaillez avec un modèle. Un modèle est un morceau de programme réutilisable. Idéalement, un professionnel devrait posséder une bibliothèque de modèles correspondant

aux interactions les plus fréquentes et à son 'look&feel' personnel. Un modèle est un bout de programme que vous sauvez comme tel (menu 'File', option 'save in model'). Attention, ce modèle doit être sauvé dans le sous-répertoire “Knowledge Objects” du répertoire “Authorware” de votre machine. Il est ensuite possible de l’appeler depuis votre nouveau programme (menu 'Window', option 'Knowledge Objects'). L'avantage de cette approche est qu'Authorware prendra en charge les éventuels conflits entre variables portant le même nom: il vous demandera de trancher s'il s'agit effectivement de la même variable ou si vous voulez renommer l'une d'entre elles afin d'éviter les confusions.

Exercice 8-9: Modèle de question dichotomique

Créez un modèle "question vrai ou faux". Ensuite utilisez-le pour réaliser un QUIZ comprenant 5 questions de type vrai ou faux. Sauvez le modèle sous 8.9.M et le quizz sous 8.9.E.

Leçon 9. Interactions complexes

9-1 Objectif

L'utilisation de variables dans la construction des interactions va vous permettre de réaliser des interactions plus complexes que les précédentes, notamment grâce à l'utilisation des réponses de type 'condition'.

9-2 Traiter les réponses de type 'texte'

Comment fournir un feed-back du type "Très bien, Christine!" ? Il faut demander le prénom de l'utilisateur, stocker la réponse dans une variable, disons la variable 'Prenom' et afficher le texte "Très bien {Prenom}!". Lorsque l'utilisateur introduit une réponse de type 'texte', celle-ci est automatiquement stockée dans la variable-système 'EntryText'. Cette variable EntryText est remise à jour à chaque interaction, c'est pourquoi il faut en copier le contenu dans une autre variable (ici 'Prenom') qui ne sera plus modifiée.

Inversement, vous pouvez donner une valeur par défaut à la réponse-texte et l'afficher dans la zone-réponse. Associez à l'icône d'interaction une instruction de type 'PresetEntry := "blabla"'. Si l'utilisateur presse RETURN sans autre, cette réponse par défaut sera traitée par la machine comme s'il en avait lui même introduit les caractères “blabla”. Si l'utilisateur presse une touche, la réponse par défaut est effacée et la réponse fournie par l'utilisateur est conservée.

Exercice 9-1: Douanier sympathique

Programmez un douanier sympathique. Il pose la question "De quel pays êtes-vous?". Par défaut, la réponse est 'Suisse'. Si la réponse de l'utilisateur est 'Suisse', donnez le feed-back "Bienvenue chez vous.". Si la réponse est un autre pays, par exemple Japon, le douanier affiche le message "Les citoyens du Japon sont les bienvenus en Suisse". Si l'utilisateur n'a pas mis de majuscule au nom du pays, assurez-vous que celui-ci commence par une majuscule au moment d'afficher le nom du pays.

9-3 Traitement des réponses numériques

Imaginons que vous demandez à l'utilisateur l'altitude du Mont-Blanc. Vous pouvez créer une réponse de type 'texte' en prévoyant le pattern "4807". Il serait cependant intéressant de donner un feed-back pour les réponses comprises entre 4800 et 4850. Si vous définissez le pattern "48??", Authorware acceptera 4801 et 4866, mais aussi 4699 ou 46mm! Le seul véritable moyen d'analyser ces réponses, c'est de faire des comparaisons arithmétiques (=, <, >,...). Malheureusement, celles-ci ne sont pas prises en charge par le processus d'analyse de réponse-texte. C'est pourquoi nous utiliserons des réponses de type 'condition' qui permettent d'évaluer un prédicat (comme les conditions étudiées dans la leçon 6): N > 34, N <= 25, ...

Aussi, le traitement des réponses numériques comporte 2 étapes: primo, accepter une réponse et secundo l'analyser:

- **Accepter** la réponse consiste à créer une interaction, comprenant une réponse de type 'texte' pour laquelle vous acceptez toute réponse (pattern = *). Authorware réalise une pré-analyse de la réponse au cours de laquelle le premier nombre identifié est stocké dans la variable-système 'NumEntry'. Pour cette réponse, sélectionnez l'option de branchement '**continue**': Authorware redessine l'ordinogramme pour montrer qu'après avoir réalisé le test 'pattern=*', il va considérer les autres tests placés sur l’axe horizontal d'analyse de réponses.
- **Analyser** la réponse exige une réponse de type 'condition'. En éditant cette icône, un seul champs nous intéresse pour le moment: 'match if true'. Vous pouvez y rédiger une condition en utilisant la

même syntaxe que pour rédiger les conditions dans une icône de calcul. Par exemple, "NumEntry >5". Rappelez-vous qu'une condition est une question: elle sert à tester une valeur et non à affecter une valeur.

La particularité des réponses numériques est de demander ce double traitement, une réponse 'texte' pour permettre à l'utilisateur d'introduire sa réponse, plus un certain nombre de réponses 'condition' pour évaluer cette réponse. En dehors de cela, ce type d'interactions fonctionne exactement comme les interactions vues précédemment. Je vous recommande notamment de placer les différentes réponses conditionnelles selon la recette communiquée précédemment:

1. Une condition définissant la ou les réponses correctes
2. Vérifier si l'utilisateur a atteint le nombre d'essais maximal défini.
3. Une condition pour chaque mauvaise réponse prévue.
4. Une condition générale pour tout autre mauvais réponse (feed-back d'erreur générique)

Le test lié au feed-back générique doit être toujours vérifié (tautologie): il s'agit par exemple du pattern universel '*' pour une réponse 'texte', ou d'une zone sensible couvrant l'écran entier. Dans le cas présent, il est facile d'écrire une condition toujours vérifiée telle que '1 = 1'. Vous pouvez aussi utiliser comme condition la constante TRUE.

Exercice 9-2: Altitude du Mont-Blanc

Demandez au sujet quelle est l'altitude du Mont-Blanc. La bonne réponse est 4807. Acceptez une erreur de 100 mètres, mais précisez alors la réponse exacte. Si l'altitude est trop faible, affectez le feed-back “cherchez plus haut”. Si sa réponse est trop élevée, affectez le feed-back “visez plus bas”. Prévoyez un feed-back d'erreur générique. A la troisième erreur, donnez la réponse correcte

Si la réponse contient plusieurs nombres, le second est stocké dans 'NumEntry2' et l'éventuel troisième nombre dans 'NumEntry3'. La variable 'NumCount' indique combien de nombres ont été identifiés dans la réponse du sujet. Un nombre est une suite ininterrompue de chiffres. Dans la réponse '1 200', le système identifiera deux nombres 1 et 200, sauf si l'auteur a demandé au pattern matcher d'ignorer les blancs.

Exercice 9-3: Trois nombres croissants

Demandez au sujet d'introduire trois nombres dans un ordre croissant. Les trois nombres sont introduits en une fois, l'utilisateur tapant par exemple "55 88 654". Limitez à 3 essais. Prévoyez le cas où l'utilisateur introduit moins de trois nombres. Si les nombres ne sont pas dans l'ordre croissant, prévoyez un feed-back du type "19 devrait se trouver avant 45 et non après."

La condition sur les nombres peut se combiner avec un test sur le texte de la réponse. Dans ce cas, faites une analyse de type texte et mémorisez son résultat dans une variable. Reprenons, par exemple, l'exercice 8.2 et imaginons que vous demandiez à l'utilisateur de fournir les unités. Testez le pattern 'm?tre*'. Si ce test est satisfait, placez comme feed-back l'affectation 'Unite := "m" '. Reprenez ensuite cette information dans les conditions. La condition définissant la réponse parfaite sera 'NumEntry = 4807 & Unite ="m" '. La condition 'NumEntry = 4807 & UniteOk ="" ' donnera par exemple lieu au feed-back "La réponse est exacte, mais vous avez oublié les unités."

Exercice 9-4: Multiplication de décimètres

Posez la question "3 dm X 10 =". Accepter la réponse '30' si l'unité fournie est 'dm', '3' si l'unité fournie est 'm' et '300' si l'unité fournie est 'cm'.

9-4 Traitement des réponses à une question comprenant des variables

Les réponses de type 'condition' permettent aussi d'analyser des réponses lorsque la question est elle-même construite au moyen de variables. Considérons le cas classique des tables de multiplication. Vous générez au hasard deux nombres N1 et N2 que l'élève doit multiplier. Dans ce cas la réponse correcte ne sera pas identifiée par une condition du type 'NumEntry = 4807' mais par une comparaison de variables 'NumEntry = N1 * N2'.

Exercice 9-5: Les (inévitables) tables de multiplication

Créez un programme d'entraînement aux tables de multiplication, dans lequel les calculs sont générés aléatoirement. En cas d'erreur, le programme dessine N1 lignes de N2 pions et demande au sujet de répondre à nouveau.

L'utilisation de variables ne se limite pas aux réponses numériques. Toutefois, lorsqu'il s'agit de questions dont la réponse ne peut être générée par des fonctions Authorware, par exemple la conjugaison des verbes ou le pluriel des noms, il faut stocker dès le départ la question et la solution. Nous en reparlerons dans la leçon 12.

Dans l'analyse de réponse 'texte', vous ne pouvez malheureusement pas définir un pattern au moyen d'une variable. Aussi, faut-il procéder comme dans le traitement des réponses numériques: primo, accepter toute réponse, secundo, tester si le pattern appartient à la réponse au moyen de la condition suivante:

```
find (pattern, entrytext) > 0          -- pattern est une variable contenant la réponse attendue
```

Exercice 9-6: Le pluriel des noms

Créez un programme d'entraînement au pluriel des noms. Le programme choisit un nom au hasard parmi les 10 noms disponibles (sans fournir deux fois le même) et demande à l'élève d'en donner le pluriel. La réponse par défaut est le nom singulier: l'utilisateur n'a ainsi qu'à modifier la terminaison, ce qui évite les erreurs de recopie du mot. En cas d'erreur, prévoir un feed-back spécifique selon la règle de pluriel: (1) règle générale (+s), (2) pluriel des mots en -al (-aux), (3) pluriel des mots en -s (-s) et (4) noms au pluriel irrégulier.

9-5 Traitement des réponses multiples

Nous traitons ici les situations dans lesquelles le sujet doit fournir plusieurs éléments de réponse: entrer plusieurs mots, cliquer dans plusieurs zones, déplacer plusieurs objets. Nous avons déjà abordé le cas des réponses 'texte' multiples, qu'Authorware prend en charge grâce à l'option 'incrémental matching'. Pour les autres type de réponse, si l'interaction comprend plusieurs actions (clicks ou ‘drag & drop’), vous devez prévoir un **signal explicite d'émission de réponse**, par exemple un bouton 'OK', 'STOP' ou encore 'J'ai terminé'.

Nous distinguons les situations où un traitement immédiat des réponses est effectué de celles où un traitement différé est réalisé. Le **traitement immédiat** consiste à traiter chaque réponse partielle avant que la réponse suivante ne soit introduite. Dans ce cas, les réponses sont en général considérées indépendamment les unes des autres. Après toute réponse, le sujet est invité à fournir une nouvelle réponse (option 'try again'), sauf lorsqu'il sélectionne le signal de fin de réponse (OK, stop,...).

En cas de **traitement différé**, les choix effectués par l'utilisateur sont mémorisés dans une ou plusieurs variables ad hoc. En général, on prévoit dans ce cas deux modes de sortie de l'interaction: une sortie avec traitement des réponses fournies (en général, bouton **'OK'**) et une sortie sans traitement des réponses fournies (en général, bouton **'cancel'**). Le traitement différé permet de considérer l'ensemble des réponses fournies comme les composantes d'une seule réponse globale. Dans ce cas, l'analyse de réponse est plus complexe car il peut être nécessaire d'analyser les relations entre les éléments de réponse. Voici 3 exemples de situa-

tions dans lesquelles une telle analyse se justifie:

- si le sujet sélectionne des options d'impression, il faut vérifier si ces options sont compatibles;
- si le sujet introduit plusieurs données, il convient de vérifier leur cohérence (par exemple, dans un formulaire, la date de naissance est-elle compatible avec le numéro de pension?);
- si le sujet doit répartir en deux ensembles cohérents un jeu de quatre objets, il convient de vérifier si les objets placés dans une même zone ont en commun des propriétés que ne possèdent pas les objets de l'autre zone.

Dans les situations didactiques, deux critères sont souvent utilisés dans l'analyse de réponses complexes: 1) est-elle complète et 2) contient-elle des éléments incorrects? Authorware permet de définir individuellement chaque réponse comme étant correcte, incorrecte ou neutre. Pour chaque interaction, il met à jour trois variables dans lesquelles ces éléments sont comptabilisés:

- dans 'CorrectChoicesMatched' se trouve le nombre de réponses fournies par l'utilisateur qui correspondent aux réponses définies par l'auteur comme correctes;
- dans 'WrongChoicesMatched' se trouve le nombre de réponses fournies par l'utilisateur qui correspondent aux réponses définies par l'auteur comme incorrectes;
- la variable 'AllCorrectMatched' vaut True si toutes les réponses définies comme correctes ont été fournies par l'utilisateur.

Lorsque le sujet fournit le signal de fin de réponse (OK, stop,...), il convient d'évaluer la valeur de ces variables (au moyen de réponses de type 'condition'). En combinant les deux dernières variables, il est possible de distinguer plusieurs classes de réponses:

- réponses complètes (tous éléments désirables: AllCorrectMatched = True) et correctes (aucun élément indésirable: WrongChoicesMatched = 0);
- réponses complètes (tous éléments désirables: AllCorrectMatched = True), mais incorrectes (présence de certains éléments indésirables: WrongChoicesMatched > 0);
- réponses incomplètes (absence de certains éléments désirables: AllCorrectMatched = False), mais correctes (aucun élément indésirable: WrongChoicesMatched = 0);
- réponses incomplètes (absence de certains éléments désirables: AllCorrectMatched = False) et incorrectes (présence de certains éléments indésirables: WrongChoicesMatched > 0);

Si l'auteur désire identifier plus finement le nombre d'éléments corrects et incorrects, par exemple pour discriminer le feed-back selon le nombre d'items manquants, il utilisera les valeurs exactes des variables 'CorrectChoicesMatched' et 'WrongChoicesMatched'. La gestion de ces variables par Authorware souffre cependant d'un inconvénient. En effet, ces variables ne sont pas remises à zéro en cours d'interaction. Dans le cas, où le sujet fournit une réponse comprenant deux éléments incorrects et qu'il lui est demandé de recommencer, la variable WrongChoicesMatched conservera la valeur 2 même s'il n'introduit pas de nouvel élément incorrect. L'auteur n'a pas la possibilité de modifier les valeurs des variables du système. D’autres solutions sont cependant possibles telles que:

- comptabiliser les éléments incorrects dans une variable créée par l'auteur, incrémentée à chaque erreur et réinitialisée en cas de nouvel essai;
- stocker (après chaque essai) la valeur de 'WrongChoicesMatched' dans une variable N; l'essai suivant, il suffit de vérifier si la nouvelle valeur de 'WrongChoicesMatched' dépasse ou non sa valeur précédente.

Exercice 9-7: Algèbre

Demander au sujet de choisir les expressions équivalentes à '3 (x+2) - 3 = 9' au sein de la liste suivante:

a) 3x+6 = 6

b) 3X-6 =6

c)3X+6 =12

d)3((x+2)-1) =9

e)(x+2) = 3

L'utilisateur répond en utilisant des réponses 'check box': il fournit une réponse à la fois et clique sur un bouton 'j'ai fini' lorsqu'il a fini. Si le sujet veut corriger sa réponse, il clique sur le bouton 'effacer', ce qui efface toutes les marques dans les check boxes et lui offre la possibilité de fournir une nouvelle réponse.

La bonne réponse est constituée par la troisième ET la quatrième expression. Fournir un feed-back pour: réponses complètes sans élément incorrect; réponses incomplètes sans élément incorrect; réponses complètes plus élément incorrect; réponses incomplètes plus élément incorrect et réponses contenant des éléments autres que a, b, c, d ou e.

Leçon 10. Interactions perpétuelles

10-1Objectif

Les fonctionnalités d'Authorware que vous avez étudiées jusqu'ici privilégient la construction de programmes de type **'scénario'**: l'utilisateur suit un scénario composé des différentes interactions que vous avez construites. Les scénarios les plus simples sont des séquences linéaires d'activités, les scénarios plus complexes comprennent des branchements et des boucles. Cette approche de type 'scénario' est particulièrement adaptée à la conception classique des didacticiels, à savoir une séquence de questions et de réponses.

Comment cependant réaliser un jeu d'échecs ou un petit logiciel de dessin avec Authorware? En effet, ces logiciels ne reposent pas sur un scénario, il s'agit plutôt d'une **'situation'** qui évolue éventuellement mais offre plus ou moins les mêmes possibilités d'interactions: les mêmes mouvements sont disponibles aux échecs en début ou fin de partie (pour autant qu'ils soient légaux), les mêmes outils sont disponibles dans un logiciel de dessin après 5 ou 55 minutes d'utilisation.

Pour construire des situations, Authorware dispose de structures d'interactions dites 'perpétuelles', c'est-à-dire pour lesquelles l'utilisateur peut fournir une réponse à tout moment. Dans cette leçon, vous apprendrez à créer des boutons perpétuels, des zones sensibles perpétuelles, etc. Nous étudierons en particulier les réponses de type 'menu déroulant' qui sont par nature des réponses perpétuelles.

10-2Les réponses perpétuelles de type 'bouton'

Dans de nombreux cas, il est intéressant d'offrir un bouton disponible pendant plus d'une interaction: 'aide', 'consignes', 'retour au menu', 'score', 'quitter'. Ce bouton est créé dans une icône d'interaction et reste disponible tout au long du programme. Il vous suffit pour cela d'éditer le bouton et de choisir l'option 'perpetual'.

La seule option propre à une réponse perpétuelle concerne le mode de branchement. En effet, si l'utilisateur clique sur le bouton perpétuel 'score' alors qu'il est à 534ième icône du programme, alors que le bouton a été créé dans la première icône, Authorware exécute le feedback associé au bouton (présenter le score), puis reprend l'exécution à la 534ième icône. Pour cela, choisissez l'option **'return'** (au lieu de 'exit interaction ' ou 'try again'). Authorware retournera à la 534ième icône dans son état exact, par exemple, si le sujet en est au troisième essai dans une interaction, Authorware s'en souvient.

Exercice 10-1: Score

Reprenez l'exercice 8.9 et sauvez-le sous 10.1. Ajoutez un bouton permanent qui permet à tout moment à l'utilisateur de connaître le nombre de réponses correctes et incorrectes qu'il a fournies.

Vous connaissez déjà l'option **'active if true'** qui permet de désactiver un bouton, mais nous vous en rappelons le fonctionnement car celui-ci est particulièrement utile pour les boutons perpétuels. Si, dans ce champ, vous exprimez une condition au moyen de variables, le bouton sera 'actif' tant que la condition sera vérifiée. Par exemple, pour désactiver un bouton 'passer au niveau inférieur' lorsque l'utilisateur a atteint le niveau le plus bas, la condition serait par exemple 'niveau > 0'. Si la variable 'niveau' n'est pas supérieure a 0, le bouton apparaîtra soit en gris (option 'dim') soit ne sera pas visible (option 'hide').

Exercice 10-2: Joker

Reprenez l'exercice 10.1 et sauvez-le sous 10.2. Ajoutez un bouton 'Joker' qui présente un indice pour chaque question vrai-faux. Au total, l'utilisateur a le droit de demander deux fois le joker. Désactiver le bouton lorsqu'il a utilisé ses deux jokers.

Les boutons de navigation dans un logiciel ou dans un hyperdocument sont généralement des boutons perpétuels. Il est possible de bricoler ces boutons. Toutefois, Authorware vous offre un jeux de boutons de navigation plus efficaces (voir leçon 11).

10-3Les réponses perpétuelles de type 'menu'

Les réponses perpétuelles les plus utilisées sont les réponses de type 'menu'. Clarifions en premier lieu l'utilisation du terme menu. Un menu est toute liste d'items au sein de laquelle l'utilisateur doit faire un choix. En d'autres termes, un menu est une question à choix multiple. Les logiciels comprennent généralement deux types de menu:

- Les menus '**pop up**' présentent les items dans une boîte de dialogue qui apparaît à l'écran à un moment précis et au sein desquels l'utilisateur doit faire un choix afin de continuer l'exécution du logiciel. Pour réaliser un tel menu il suffit de créer une icône d'interaction comprenant ces différents items sous forme par exemple de boutons radio ou de check-boxes.
- Les menus '**pull down**', en français 'menus déroulants', sont disponibles en permanence dans la barre des menus située en haut de l'écran. Pour les créer, Authorware vous offre des réponses de type menu.

Pour créer un menu déroulant, créez une icône d'interaction. Le nom de l'icône d'interaction sera le nom du menu tel qu'il apparaîtra dans la barre de menu. Les items du menu seront les différentes réponses de type 'menu' que vous aurez associées à cette icône d'interaction.

L'édition d'une réponse de type 'menu' comporte différents paramètres que vous connaissez déjà. Vous choisirez en générale les options 'perpetual' et 'return'. L'option 'Optional Key' permet de créer des raccourcis-clavier: par exemple, si ce champ comporte la lettre G, le feed-back sera déclenchée soit si l'utilisateur sélectionne cet item dans le menu, soit s'il presse CTRL-G sur Windows.

Exercice 10-3: Menu déroulant

Reprenez l'exercice 10.2 et sauvez-le sous 10.3. Remplacez les boutons permanents par des items au sein d'un menu "Jeu".

En réalisant cet exercice, vous aurez noté que:

- Un item désactivé apparaît automatiquement en gris, vous n'avez pas la possibilité de le faire disparaître comme pour les boutons permanents.
- Les menus Authorware disparaissent en mode exécution, afin d'afficher les vôtres. Ceux-ci apparaissent de gauche à droite selon l'ordre où ils sont créés dans votre programme.
- Par défaut, Authorware crée un menu 'file' comprenant uniquement l'option 'quit'. Si vous voulez ajoutez des options à FILE, créez vous-même un menu file en y ajoutant vos options.
- Authorware ne permet pas de créer des menus arborescents, c'est-à-dire comprenant des sous-menus (sauf en appelant des fonctions externes).

Pour structurer le menu, vous pouvez insérer (sur la version Windows) un des codes suivants devant le nom de l'item dont il désire modifier l'apparence:

(-	pour laisser un traitillé dans le menu (séparateur).
(	pour laisser une ligne vide dans le menu.
&	pour souligner l'item dans le menu.

10-4Réponses conditionnelles perpétuelles

Si vous créez une réponse conditionnelle perpétuelle, Authorware déclenchera celle-ci chaque fois que la condition sera vérifiée. Ceci vous permet - dans une certaine mesure - de faire de la programmation à base d'événements ('**event-driven programming**'). Il s'agit de créer des éléments de programme qui ne s'enchaînent pas séquentiellement, mais se déclenchent en fonction de certains événements. Un événement est ici défini à partir d'une condition. Par exemple, une réponse conditionnelle perpétuelle "score > 10" se déclenchera lorsque cette condition sera satisfaite. Certes, cette réponse conditionnelle pourrait ne pas être perpétuelle, mais simplement se trouver dans une icône d'interaction. Par contre, si le score est comptabilisé à travers différentes icônes d'interaction, la condition perpétuelle sera détectée lorsque la valeur de score dépassera 10, quel que soit l'endroit du programme.

Exercice 10-4: Joker épuisé

Modifiez le programme 10.3 et sauvez-le sous 10.4. Lorsque l'utilisateur a épuisé ses jokers, un message apparaît pour l'informer que celui-ci n'est plus disponible.

10-5Autres réponses perpétuelles

Vous pouvez créer des réponses perpétuelles de type zone sensible, objet sensible et objet mobile. Celles-ci fonctionnent comme les réponses perpétuelles de type bouton.

Par contre, vous ne pouvez créer de réponse perpétuelle de type texte, car pour ce type de réponse, Authorware s'interrompt et attend que l'utilisateur fournisse une réponse. Il n'existe pas non plus d'option 'perpétuelle' pour les réponses de type touche, temps de réponse et nombre d'essais.

10-6Gestion du temps

Imaginez la situation suivante. Vous avez conçu un exercice d'orthographe au sein duquel l'élève dispose d'un temps de réponse maximal de 60 secondes. Il dispose aussi de trois boutons permanents: 'Grammaire' pour consulter les règles mis en jeu dans l'exercice, 'Score' pour voir son score et 'Autre phrase' pour changer la phrase proposée, par exemple parce que l'élève n'en maîtrise pas le vocabulaire. Que se passe-t-il s'il consulte la grammaire ou son score pendant 15 secondes ? Faut-il décompter ces 15 secondes des 60 secondes allouées ou non ? Authorware vous offre quatre possibilités, sous le menu "If interrupted by perpetual interaction":

- 'Continue timing' permet de continuer le décompte pendant l'interaction perpétuelle. Il est par exemple logique que le décompte du temps continue pendant la consultation de la grammaire puisque cette activité fait partie du raisonnement qui conduit à la réponse.
- 'Pause, Resume on Return' suspend le décompte pendant l'interaction permanente et le reprend au moment où celle-ci se termine et que l'utilisateur revient au point où il était. Cette option s'appliquerait ici au cas où le sujet consulte son score, car cette opération est neutre par rapport à la tâche (attention toutefois aux tricheurs!).
- 'Pause, Restart on Return' oublie en quelque sorte le temps écoulé avant la réponse permanente. Le compte à rebours est repris à son point de départ lorsque l'élève revient dans l'interaction. Dans l'exemple ci-dessus, si l'élève choisit 'autre phrase', il disposerait à nouveau du temps total, soit 60 secondes.
- Imaginons que l'élève ait déjà dépassé le temps de réponse fixé et qu'il n'ait pas quitté l'interaction, par exemple parce que le feed-back associé à ce temps maximal ne consistait pas à donner la bonne réponse mais seulement à fournir un indice. Ensuite, cet élève utilise le bouton permanent 'Autre phrase'. Lorsqu'il revient à la question, faut-il recommencer le compte à rebours, bien que celui-ci soit déjà terminé ? Si vous choisissez l'option ci-dessus 'Pause, Restart on Return', le compte à rebours sera reinitialisé même s'il était préalablement terminé. Par contre, l'option 'Pause, Restart if Running' précise que le compte à rebours ne doit être recommencé que si celui-ci n'avait pas été

mené à terme. Si le décompte sert à fournir un indice, il n'est pas utile de le recommencer, puisqu'il est inutile de fournir deux fois le même indice. Par contre, si la phrase a changé, il est pertinent de reprendre le compte à rebours précédant la présentation de l'indice.

Exercice 10-5: Joker sur le temps

Modifiez le programme 10.4 et sauvez-le sous 10.5. Fixez un temps de réponse maximal de 10 secondes. Après ce délai, donnez la bonne réponse. Lorsque l'utilisateur joue un de ses jokers, quel que soit le temps écoulé, les 10 secondes lui sont à nouveau accordées.

10-7Hygiène de programmation

Attention, vous ne savez pas à l'avance quand l'utilisateur cliquera sur un bouton permanent, c'est-à-dire quel est l'état de l'écran avant qu'il sélectionne un bouton permanent. En réalité, il n'est pas nécessaire de le savoir tant que les instructions exécutées suite à la sélection du bouton permanent n'ont pas d'effet sur ce qui était préalablement affiché. Par contre, si le feed-back du bouton perpétuel modifie l'état préalable de l'écran, par exemple s'il en efface une composante, Authorware ne reconstituera pas celui-ci à son retour. Donc, si vous désirez qu'à son retour, l'utilisateur trouve l'écran dans son état au moment du départ, il faut concevoir les feed-back des boutons permanents comme des composantes hermétiques:

- aucune icône du feed-back du bouton permanant n'efface les éléments affichés préalablement;
- les éléments du feed-back du bouton permanant se superposent au contenu de l'écran, le masquent éventuellement, mais sont tous effacés en fin de feed-back.

Cette recette n'est bien entendu pas toujours applicable, par exemple, si vous ajoutez un bouton 'effacer écran'! Elle correspond néanmoins à une règle d'hygiène assez générale en programmation qui consiste à réaliser des bouts de programme (procédures, routines, fonctions, modules, ...) les plus autonomes possibles les uns par rapport aux autres. En effet, si le feed-back d'un bouton permanent n'a aucune interférence avec l'état préalable de l'écran (par exemple, un élément de l'écran préalable rend illisible le texte du feed-back), il suffit de tester ce feed-back une fois pour toutes. Par contre, s'il y a interférence, vous devrez tester le feed-back pour chaque endroit depuis lequel il peut être appelé.

Outre les interférences graphiques, vous risquez de rencontrer des interférences sur des variables lorsque: le programme du bouton permanent change la valeur d'une variable utilisée dans le programme, ou vice-versa. Cette interférence peut être voulue: par exemple, le bouton perpétuel 'plus vite' modifie la variable 'vitesse' utilisée par les animations du programme ou le bouton 'calculs moins difficiles' modifie les variables qui déterminent la génération des calculs. Par contre, si cette interférence n'est pas explicitement voulue, une règle générale consiste à utiliser des noms différents pour des variables utilisées à différents endroit du programme. D'autres langages de programmation disposent des variables '**locales**', de telle sorte que, même si elles portent le même nom, leur valeur dans une procédure n'affecte pas leur valeur dans une autre procédure. Authorware utilise des variables locales pour certaines icônes, par exemple la variable tries@"question1" indique le nombre d'essais effectués à l'interaction "question1" et n'affecte par exemple pas la valeur de tries@"question2". Il en est de même des variables utilisées par Authorware dans les icônes de décision, d'animation, etc. Par contre, **les variables que vous créez sont des variables globales**. Chaque variable a une valeur unique, quel que soit le point du programme où elle est appelée.

Leçon 11. La navigation dans les hyperdocuments

11-1Objectif

Dans cette leçon, vous apprendrez à créer un **hyperdocument**, c'est-à-dire un document dont la structure n'est pas linéaire. Dans un document linéaire, les pièces d'information se succèdent dans un ordre prédéterminé, telles les pages d'un livre. Dans un hyperdocument, chaque pièce d'information est connectée à un certain nombre d'autres pièces d'information. Les connections offertes sont présentées au lecteur qui choisit celles dont il désire prendre connaissance en premier lieu. Un hyperdocument est un graphe de type 'réseau sémantique[54]' dont les éléments sont appelés des **noeuds** et les relations entre éléments sont appelés des **liens**. Un lien X-Y signifie que l'élément X (un mot, un objet graphique,...) d'un noeud est lié au noeud Y. Cet élément X doit être un objet activable (un bouton, une zone sensible,...) de telle sorte que, si l'utilisateur sélectionne X, on lui présentera le noeud Y.

Ces notes de cours présentées sur WWW constituent un exemple d'hyperdocument. Nous étudions ici comment réaliser un hyperdocument avec Authorware. Les possibilités d'intégration entre Authorware et HTML évoluent rapidement. Elles ne sont pas décrites dans ce document. Le lecteur est invité à se référer aux pages WWW.

Le noeud est l'unité de base de l'hyperdocument. Il peut comporter du texte, mais également des graphiques, des images vidéo, du son. Un noeud peut même être un petit programme avec lequel l'utilisateur travaille sur un point particulier. La terminologie pour décrire un noeud varie d'un système à l'autre. On parle d'**hypertexte** lorsque les noeuds du graphe sont des textes et d'**hypermedia** lorsque les noeuds comportent des images ou du son.

Les liens d'un hyperdocument ne respectent pas la structure d'un programme Authorware. En effet, ce langage repose sur une logique d'emboîtement, de boîtes comprises dans d'autres boîtes, qui est a priori incompatible avec une structure “HYPER” dans laquelle n'importe quel élément est susceptible d'être connecté avec n'importe quel autre. Il est certes possible de bricoler une solution au moyen des composantes d'Authorware étudiées jusqu'ici: des boutons permanents, des fonctions de type 'saut', etc. Toutefois, comme il s'agit de fonctionnalités fréquemment utilisées[55], les concepteurs d'Authorware ont ajouté depuis la version 3 de nouvelles fonctionnalités pour créer un hyperdocument et gérer la navigation de l'utilisateur dans cet hyperdocument.

11-2Créer la structure d'un hypertexte: icône framework

Un hyperdocument comporte d'une part, un ensemble de noeuds, créés indépendamment les uns des autres, et 'mis à plat' dans une base de données, et d'autre part, les modalités de navigation entre ces noeuds. Un noeud est généralement une icône de présentation avec du texte et/ou du graphisme. Un noeud peut aussi contenir des animations, du son ou des interactions. Chaque noeud est un sous-programme indépendant des autres noeuds du programme.

Je vous propose de créer ensemble un premier hypertexte simple, dans lequel l'utilisateur peut faire dérouler les pages.

---

54. Un réseau sémantique est la représentation d'un ensemble de concepts par un graphe dans lequel chaque concept est représenté par un point et les relations entre deux concepts (exemple-de, contraire-de, propriété-de, ...) sont représentées par les arcs entre ces points.

55. Même si le logiciel que vous développez n'est pas un hyperdocument, il est fort probable que ces fonctionnalités vous soient utiles pour une composante particulière du programme, par exemple, la rédaction d'une documentation on-line.

Exercice 11-1: Hypertexte 1

Premier pas: créez les noeuds

*Nous travaillons sur un hyperdocument simple comprenant quatre **noeuds** (ou pages), affichant chacun une forme géométrique. Placez dans le programme quatre icônes de présentation affichant respectivement un disque, un carré, un triangle, et une ellipse de différentes couleurs.*

Deuxième pas: créez la base de données

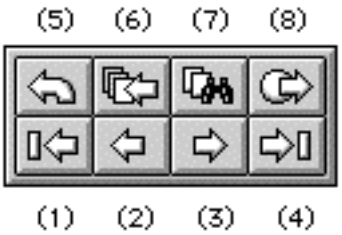
*Posez sur la ligne de déroulement du programme une icône **'framework'.** Placez ensuite chacun des noeuds à droite de l'icône de framework, comme pour placer les feed-back dans une icône d'interaction.*

Troisième pas: exécutez

Faites un run. Vous verrez apparaître le premier noeud de votre hyperdocument ainsi qu'une palette comprenant 8 boutons de navigation. Cliquez sur la flèche vers la droite pour faire défiler les pages. Bravo, vous venez de créer un hyperdocument! Utilisez les autres boutons de navigation pour en deviner les effets.

D'où vient cette palette de navigation qui est apparue dans l'exercice 11.1? Si vous double-cliquez dans l'icône 'framework', Authorware ouvre une fenêtre scindée horizontalement en deux parties, 'entry' (partie supérieure) comprend quelques icônes et 'exit' (partie inférieure') est vide. Ces deux parties comprennent des icônes qui seront exécutées respectivement lorsque le programme arrive à l'icône framework et lorsqu'il en sort.

Ce sont les icônes de la partie 'entry' qui élaborent l'outil de navigation. Il s'agit en fait d'une icône d'interaction comprenant 8 boutons permanents. Le feed-back de chaque bouton est une icône de navigation. C'est elle qui sélectionne quelle icône parmi celles présentes dans le framework doit être affichée. Les 8 boutons disponibles par défaut correspondent aux besoins standards:



1. Retour vers le premier noeud du framework
2. Retour vers le noeud précédent du framework
3. Saut vers le noeud suivant dans le framework
4. Saut vers le dernier noeud du framework
5. Retour vers le dernier noeud consulté. Attention ce bouton n'a pas la même fonction que le (2). Le bouton (2) affiche le noeud qui se trouve juste devant le noeud actuel dans l'icône framework, même s'il n'a jamais été lu (par exemple, si l'utilisateur explore le document à reculons). Le bouton (3) lit le dernier noeud que l'utilisateur a consulté. Comme il s'agit de documents à explorer dans un ordre quelconque, il peut s'agir de tout autre noeud.
6. Le bouton (6) affiche la liste des noeuds consultés par l'utilisateur (on parle aussi de la **trace** de son cheminement dans l'hypertexte). Celui-ci clique ensuite sur le noeud vers lequel il veut retourner.
7. Permet à l'utilisateur de demander au programme de rechercher un mot dans l'hypertexte. Nous décrivons cette fonctionnalité plus en détail par la suite.
8. Permet de quitter le framework. Les icônes placées dans la partie 'exit' du framework seront exécutées

au moment de la sortie.

L'astuce des concepteurs d'Authorware est d'avoir créé ces fonctions de navigation au moyen des icônes traditionnelles d'Authorware, présentation et interaction. Ceci permet à l'auteur de modifier ces fonctionnalités. Vous pouvez:

- Modifier la taille, la position ou le graphisme des boutons, ainsi que la présentation de la palette (icône 'Gray navigation panel').
- Supprimer les boutons inutiles, en créer d'autres, modifier les modalités de navigation associées à chaque icône.
- Changer le type de réponse, par exemple, utiliser des réponses de type menu.

Enfin, les icônes placées dans la partie 'exit' de l'icône framework seront exécutées lorsque le programme quitte cette icône.

Exercice 11-2: Hypertexte 2

Prenez un texte quelconque disponible sous traitement de texte et faites un copier-coller vers Authorware. Découpez le texte en 10 pages et créez un hyperdocument. Placez les boutons de navigation en une seule ligne en bas de l'écran. N'utilisez pas les boutons (6) et (7) (Cf. page précédente). Ajoutez un message de bienvenue et de salutations à l'entrée et à la sortie de l'hyperdocument.

11-3Gérer la navigation

L'icône de navigation permet de préciser une destination, c'est-à-dire de spécifier quelle sera la prochaine icône à exécuter. Cinq types de destination sont offerts:

- L'option '**Recent**' permet de remonter dans la trace des icônes préalablement sélectionnées: l'option '**go back**' sélectionne la dernière icône de cette trace, l'option '**list recent pages**' affiche la trace (liste des icônes rencontrées) à l'utilisateur et c'est ce dernier qui choisit la page qu'il veut relire.
- L'option 'Nearby' permet de choisir les icônes selon leur position dans le framework, la précédente ('previous'), la suivante ('next'), la première ('first'), la dernière ('last') ou de quitter le framework ('exit framework') en exécutant au passage les icônes placées dans la partie 'exit' de l'icône framework.
- L'option 'Anywhere' permet de spécifier une page manuellement, soit parmi les pages du framework actif (son nom est affiché), soit parmi toutes les pages du logiciel ('entire file'). Vous remarquerez en particulier ici combien il est important de donner à vos icônes des noms qui en reflètent vraiment le contenu. Néanmoins, si vous ne vous souvenez plus du nom de l'icône, vous pouvez utiliser le bouton 'find' pour rechercher les icônes où figure un mot ou un mot-clé particulier. Les sauts vers des pages extérieures au framework doivent être utilisés avec la plus grande prudence.
- L'option 'Calculate' permet de générer le nom de la page à rechercher au moyen de fonctions et de variables. Il s'agit d'une option réservée aux utilisateurs avancés.
- L'option 'Search' permet à l'utilisateur de formuler lui-même le texte qu'il veut rechercher dans l'hyperdocument. La recherche peut être limitée au framework ('search current framework') ou couvrir tout le logiciel ('search entire file'). Le texte introduit par l'utilisateur peut être cherché dans le texte des icônes ou parmi les mots-clé.

En général, le saut vers la destination choisie constitue un aller-simple (option '**jump to page**'). Si le sujet veut revenir ensuite au point de départ, il dispose des boutons de navigation lui permettant de le faire. Toutefois un aller-retour (option '**call and return**') se justifie dans certains cas, par exemple lorsqu'on sort du framework ou lorsque la branche est un 'cul-de-sac'. C'est notamment le cas des noeud de type 'note': il s'agit de textes relativement cours (une définition, un exemple, ... comme une note de bas de page dans un document imprimé), que l'utilisateur consulte sans trop s'éloigner du texte.

Exercice 11-3: Hypertexte 3, titres

Modifiez le programme 11.2 et sauvez-le sous 11.3. Ajoutez entre les pages existantes deux pages de titre de chapitre, "Chapitre 1" et "Chapitre 2". Ajoutez deux boutons de navigation, un pour aller au début du chapitre 1 et un au début du chapitre 2. Supprimez le bouton "retour à la première page" s'il est redondant avec le bouton "Chapitre 1".

11-4Créer des mots-sensibles

La création de mots sensibles se fait à partir des styles de caractères (menu 'text', options 'define styles' et 'apply styles'. De manière générale, je vous recommande d'utiliser ces styles pour réaliser un hypertexte. Le principe est le même que celui les 'styles de paragraphe' utilisés dans un traitement de texte. Plutôt que d'effectuer 'à la main' la mise en page de chaque paragraphe, l'auteur définit différents styles: le texte simple, les différents niveaux de titres, les exemples, les listes, les légendes de figures, etc... L'utilisation de styles garantit l'homogénéité du document. Il facilite surtout les mises à jour ultérieures puisqu'il suffit de modifier le style pour changer la présentation du texte partout où ce style est appliqué.

Attention dans Authorware, il s'agit de **styles de caractères** plutôt que de styles de paragraphe. En effet, ils décrivent les caractéristiques typographiques (police, taille, couleur et style des caractères, plus le format d'affichage des nombres), mais pas les caractéristiques du paragraphe (alignement, espacement, marges, ...). Ceci s'explique par le fait que ces styles sont essentiellement utilisés pour créer des **invariants dans les modalités de navigation**. Par exemple, l'utilisateur sera informé dès le départ que tous les mots en italique sont des boutons de l'hypertexte. La spécificité des styles de caractères Authorware est que vous pouvez en définir l'interactivité.

Par analogie aux objets sensibles et aux zones sensibles, nous appellerons 'mot sensible' les mots qui servent de bouton dans l'hypertexte. L'analogie est d'autant plus justifiée que vous pouvez définir des paramètres semblables:

- le nombre de 'clicks' que l'utilisateur doit effectuer sur le mot pour déclencher le saut (0,1 ou 2) (C.f. leçon 2),
- la possibilité d'inverser momentanément le mot sélectionné,
- la possibilité de modifier le curseur lorsque celui-ci survole un mot-sensible[56],
- enfin, et surtout, vous pouvez définir le saut de page associé à ce bouton, au moyen de l'option 'navigate to'. Celle-ci ouvre une fenêtre de dialogue semblable à celle de l'icône de navigation expliquée ci-avant. Vous pouvez donc régler le type de saut de la même manière.

Etant donné qu'un même style s'applique à de nombreux mots sensibles, vous devrez ensuite éditer une par une la destination de chaque mot sensible. Pour cela, en mode auteur, double-cliquez sur le mot-sensible et choisissez l'option 'navigation' dans le menu 'Text'.

Exercice 11-4: Hypertexte 4, liens externes

Modifiez le programme 11.3 et sauvez-le sous 11.4. Selon le contenu de votre texte, ajoutez des mots sensibles dans le texte qui permettent de passer directement d'un paragraphe vers un paragraphe situé sur une autre page. Les mots sensibles seront affichés en gras et en vert.

Un texte peut être présenté dans une fenêtre déroulante (menu 'Text', option 'scrolling'). Si, suite à une opération de navigation, Authorware recherche un mot de ce texte et que ce mot se trouve dans la partie non-visible du texte déroulant, il fera défiler ce texte de manière à afficher directement le mot recherché. Ceci permet en outre de réaliser des sauts entre différentes parties d'un même texte déroulant, c'est-à-dire des

56. A propos de Design : Cette option permet la mémorisation des modalités de navigation. Vous désirez par exemple que les mots en gras+bleu signifient qu'une courte définition sera présentée en bas de page ('note'), alors que les mots en gras + rouge indiquent un saut vers une autre page. L'utilisateur risque d'oublier cette convention. Vous pouvez dans ce cas utiliser deux curseurs différents pour ces boutons différents, par exemple une loupe pour la définition et une flèche pour le saut de page.

sauts internes à une page.

11-5Gestion des hyperdocuments

Lorsque vous devrez travailler sur un hyperdocument, vous serez confronté à la difficulté de maintenance des liens. En effet, l'hyperdocument ressemble rapidement à un grand plat de spaghettis. Authorware vous facilite partiellement cette maintenance par différentes options:

- L'option '**connections**' du menu 'Modify/Icon' vous présente la liste des liens vers l'icône en cours d'édition, la liste des liens qui partent de cette icône et éventuellement des liens internes.
- Lorsque vous choisissez 'define styles' dans le menu 'Text', l'option '**reference**' vous indique les icônes dans lesquelles ce style a été utilisé.
- Dans le programme, un petit triangle, reproduisant en miniature l'icône de navigation, est affiché à côté des icônes contenant du texte interactif[57].
- Lorsque vous créez une destination, l'option '**anywhere**' dans le menu 'Text/Navigation' vous permet soit de spécifier simplement l'icône désirée en cliquant dessus, soit de la rechercher à partir de textes ou de mots-clé.

Exercice 11-5: Hypertext 5, table des matières

Modifiez le programme 11.4 et sauvez-le sous 11.5. Ajoutez une table des matières au document choisi.

11-6Créer des schémas contenant des boutons de l'hypertexte

Un hyperdocument ne contient pas que du texte et des mots sensibles. Vous pouvez insérer des schémas, des animations, des photographies digitalisées, etc. Ces éléments multimédia peuvent contenir des boutons de navigation dans l'hyperdocument: par exemple, si l'utilisateur clique sur un élément du schéma, il saute vers le texte correspondant à ce schéma. Pour réaliser cela, vous n'utiliserez pas les styles de texte vus ci-avant, mais placerez comme élément du framework, un sous-programme plutôt qu'un simple icône de présentation. Ce sous-programme comprend une icône d'interaction dans laquelle vous insérez des réponses de type 'objet sensible' ou 'zone sensible', dont les feed-back comprennent une icône de navigation.

Exercice 11-6: Hypertexte 6, navigation par carte

Modifiez le programme 11.5 et sauvez-le sous 11.6 Nous traitons dans ce cours des problèmes de désorientation que rencontre le lecteur d'un hyperdocument. Ces problèmes sont partiellement réduits si l'utilisateur dispose d'une représentation globale des informations de l'hypertexte. Réalisez un carte interactive, représentant les noeuds principaux de l'hypertexte, sur lesquels le sujet peut cliquer pour se déplacer dans l'hypertexte.

57. de la même manière qu'une miniature de l'icône de calcul est affichée à côté des icônes auxquelles un calcul a été associé

Leçon 12. La gestion des fichiers

12-1Objectif

Dans la plupart des applications importantes, il est nécessaire de gérer des informations, par exemple, des listes de questions ou des informations concernant les activités de l'utilisateur. Cette leçon vous explique comment stocker les informations dans des tableaux, les sauver dans un fichier (sur disque) et les lire depuis un fichier. **Attention, la gestion des tableaux de variables a été améliorée dans Authorware 5. Ce qui est décrit ci-dessous concerne Authorware 3.5.**

12-2Les tableaux de données

Vous avez appris à utiliser quelques données, lesquelles sont stockées dans des variables. Vous pouvez par exemple, conserver dans une variable le nom de la dernière action de l'utilisateur. Imaginons par contre que vous vouliez garder une trace complète de ses actions, par exemple, pour rejouer une partie ou pour chercher la cause des difficultés d'un apprenant. Vous n'allez bien sûr pas créer 600 ou 2000 variables! Pour stocker de nombreuses données en mémoire vive, on utilise un tableau de données. En général un tableau a plusieurs dimensions; malheureusement, les tableaux dans Authorware 3.5 sont unidimensionnels, il s'agit en fait de 'vecteurs'. A les différence des variables, les éléments d'un vecteur ne sont pas accessibles par leur nom, mais par leur position dans le vecteur.

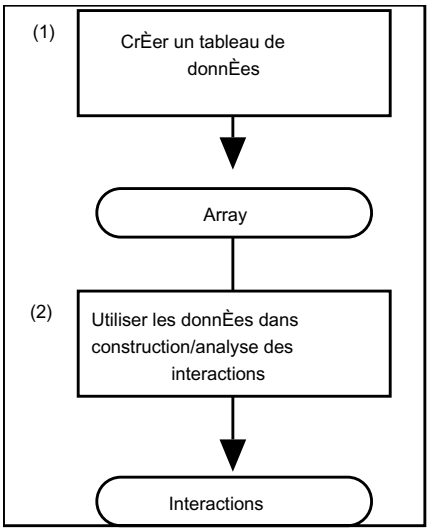


Figure 12-1:Créer et utiliser un tableau de données

La figure 12.1 explique comment générer des interactions à partir d'un vecteur de données. La création d'un tableau (étape (1) dans la figure 12-1) se fait au moyen de l'instruction ArraySet dans une icône de calcul. Celle-ci prend deux arguments, la position de la donnée et la donnée elle-même. Prenons un exemple:

```
ArraySet(1, "Belgique")
ArraySet(2, "France")
ArraySet(3, "Italie")
ArraySet(4, "Allemagne")
ArraySet(5, "Suisse")
```

Cette séquence d'instructions crée le tableau ci-dessous:

Position	*Information*
1	Belgique
2	France
3	Italie
4	Allemagne
5	Suisse

Figure 12-2:Tableau de valeurs

Au cours de l'étape (2) de la figure 12-1, vous utiliserez l'information du tableau pour construire les interactions. Cette opération consiste à rechercher une information dans ce tableau, selon sa position, et la stocker dans une variable que vous utiliserez par la suite.

`NomPays := ArrayGet(5)` La variable '`NomPays`' prend la valeur "Suisse". Laquelle était stockée à la position 5 de l'array.

Vous pouvez utiliser ce tableau pour tirer un nom au hasard:

```
NomPays := ArrayGet(random (1,5,1))
```

Vous constatez que le tableau n'a pas de nom, Authorware ne gère qu'un seul tableau. En outre, celui-ci ne possède qu'une dimension. Vous pouvez néanmoins simuler deux dimensions:

```
ArraySet(11, "Bruxelles")
ArraySet(12, "Paris")
ArraySet(13, "Rome")
ArraySet(14, "Berlin")
ArraySet(15, "Berne")
ArraySet(21, "Bruxelles-Zaventem")
ArraySet(22, "Paris-CDG")
ArraySet(23, "Milan")
ArraySet(24, "Frankfurt")
ArraySet(25, "Zurich-Kloten")
```

#	*Nom*	#	*Capitale*	#	*Principal aéroport*
1	Belgique	11	Bruxelles	21	Zaventem
2	France	12	Paris	21	Paris-CDG
3	Italie	13	Rome	23	Milan
4	Allemagne	14	Berlin	24	Frankfort
5	Suisse	15	Berne	25	Zurich-Kloten

Figure 12-3:Tableau dimensionnel de valeurs

Dans l'exemple ci-dessus, le nom du pays est à la position *i*, sa capitale à la position *10+i*, et de son aéroport principal à la position *20+i*.. On peut donc se représenter les données comme un tableau, mais il s'agit en fait d'une liste dont certaines cases sont vides (Cf. figure 12-4).

1	Belgique
2	France
3	Italie
4	Allemagne
5	Suisse
6	
7	
8	
9	
10	
11	Bruxelles
12	Paris
13	Rome
14	Berlin
15	Berne
16	
15	
18	
19	
20	
21	Zaventem
21	Paris-CDG
23	Milan
24	Frankfort
25	Zurich-Kloten

Figure 12-4:Tableau bidimensionnel de valeurs 'mis à plat' dans Authorware

Pour rechercher de l'information dans un tableau, il faut créer une boucle (Cf. leçon 6) qui explore le tableau ligne par ligne et teste chaque information. Admettons que le sujet désire connaître la capitale d'un pays, il introduit le nom du pays (celui-ci est donc stocké dans la variable 'EntryText'). Il faut alors identifier ce nom dans le tableau. La procédure ci-dessous compare ligne par ligne le tableau au nom de pays communiqué par l'utilisateur. Si le nom de pays est identifié, on regarde le nom de capitale correspondant.

```
repeat with position:= 1 to 6
if ArrayGet(position)= EntryText then
capitale:= ArrayGet(position +10)
end repeat
```

Exercice 12-1: Capitale

Choisir un pays au hasard parmi les pays de la Communauté Européenne et demander le nom de la capitale.

12-3Lire et analyser un fichier externe

Les tableaux de données permettent de créer des interactions génériques: il s'agit d'interactions dans lesquelles les questions et les réponses ne sont pas stockées dans le code même du programme mais sont soit générées au hasard, soit prélevées dans un tableau de données. Nous avons déjà étudié la génération aléatoire, ses avantages et ses limites. Si les données sont puisées dans un tableau, celui-ci peut soit être rempli dans le logiciel même, soit constitué à partir d'un fichier externe.

L'avantage de ces fichiers externes est qu'il peuvent par être édités par l'utilisateur: dans le cas d'un didacticiel, l'enseignant pourra éditer ce fichier pour ajouter des questions, des réponses, des exemples, etc. On parle dans ce cas de **didacticiel ouvert**. Permettre à l'enseignant d'adapter l'outil aux caractéristiques de sa classe augmente la probabilité que celui-ci puisse effectivement l'utiliser. Si la structure du fichier est simple, celui-ci pourra être édité avec un simple traitement de texte. Si la structure est complexe, il est préférable de créer (par exemple avec Authorware) un interface permettant d'éditer le fichier sans se soucier du format particulier des données.

Afin d'utiliser les données figurant dans un fichier externe, il est préférable de procéder en trois phases représentées sur la figure 12.2 : (1) lire les données, (2) les organiser dans un tableau et (3) les utiliser dans les interactions. Les étapes (2) et (3) on été écrites dans la section précédente. L'étape (2) n'est pas absolument indispensable, mais elle permet de mieux structurer les données et donc de rédiger un programme plus 'propre'.

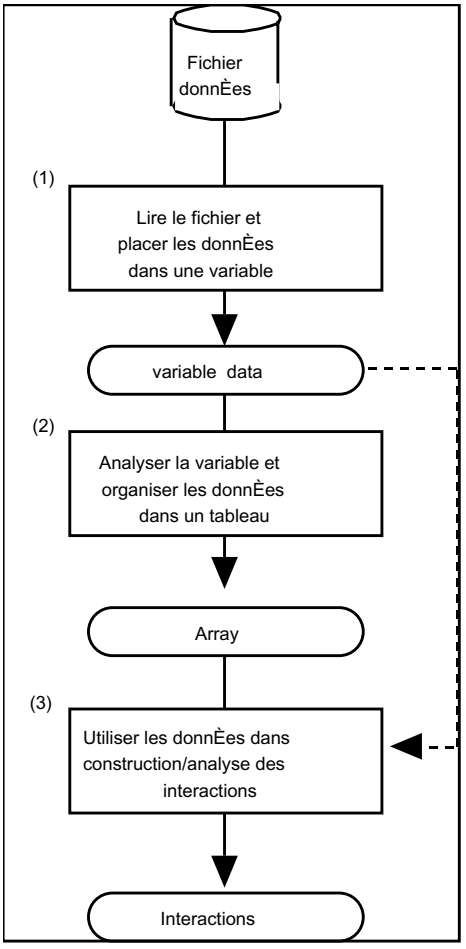


Figure 12-5:Construction d'interactions au moyen de données stockées dans un fichier externe

Lire un fichier externe dans Authorware consiste en une seule opération. La difficulté consiste à traiter ces données. En effet, comme dans l'exemple ci-dessous, les caractères qui figurent dans le fichier "Questions.txt" sont placés ensemble dans une seule variable, la variable 'data'. L'étape (1) de la figure 12.2 se

limite donc à un seule ligne:

```
Data := ReadExtFile("Questions.txt")
```

Il est préférable que ce fichier soit un simple fichier texte, c'est-à-dire qu'il ne comprenne pas d'autres caractères que le texte lui même et les sauts de ligne (RETURN). Par exemple, pour construire le tableau de données présenté en 12.3, le fichier texte pourrait être le suivant (le nom du pays est systématiquement suivi du nom de la capitale et de l'aéroport):

```
Belgique
Bruxelles
Zaventem
France
Paris
Paris-CDG
Italie
Rome
Milan
Allemagne
Berlin
Frankfort
Suisse
Berne
Kloten
```

Le programme doit alors décomposer cette longue chaîne de caractères pour y retrouver les différents éléments. Le programme de lecture du fichier consiste à placer l'ensemble des données dans la variable 'data' et à analyser ensuite ligne par ligne le contenu de cette variable. La fonction GetLine (variable, n) permet de prélever n ième ligne de la variable. La première ligne de la variable est la ligne 0. La procédure ci-dessous lira donc les lignes 0, 1 et 2 pour le premier pays et placera respectivement les informations dans les cellules 1, 11 et 21 du tableau de données. Le module (2) de la figure 12.2 prendra donc la forme suivante:

```
repeat with country := 0 to 4
    ArraySet (country, GetLine(data, country * 3))
    ArraySet (10 + country, GetLine(data, 1 + country * 3))
    ArraySet (20 + country, GetLine(data, 2 + country * 3))
end repeat
```

Une ligne pour l'instruction "GetLine" est une séquence de caractères qui se termine par RETURN. Cette ligne peut donc occuper plusieurs lignes à l'écran (il s'agit donc plus exactement d'un paragraphe).

Exercice 12-2: Quiz

Créer un QUIZ éditable. Le quiz est constitué de dix questions 'vrai - faux'. L'utilisateur répond au moyen de boutons. Les dix questions sont stockées dans un fichier externe. Pour chaque question, il faut stocker l'affirmation à présenter, suivie de 'vrai' ou 'faux'. Les questions sont présentées dans l'ordre où elles figurent dans le fichier.

Il se peut que votre logiciel ne trouve pas le fichier de questions. Vous pouvez préciser l'arbre des répertoires. En réalité, l'arbre des répertoires du fichier Authorware (votre programme) se trouve dans la variable FileLocation. Aussi, si vous concatenez celui-ci au nom de votre fichier de données et que vous placez bien ce fichier données dans le même répertoire que le programme, celui-ci le retrouvera sans peine.

```
Data := ReadExtFile (FileLocation ^ "Questions")
```

12-4Sauver les données

La sauvegarde des données sur fichier suit , mais dans le sens opposé, les mêmes étapes que leur lecture. L'étape (1) consiste à sélectionner les variables Authorware pertinentes (réponses fournies, nombre de réponses correctes, nombre d'essais, temps de réponses, choix effectués, ...) et à stocker ces réponses dans un tableau, par exemple en fonction du numéro de la question.

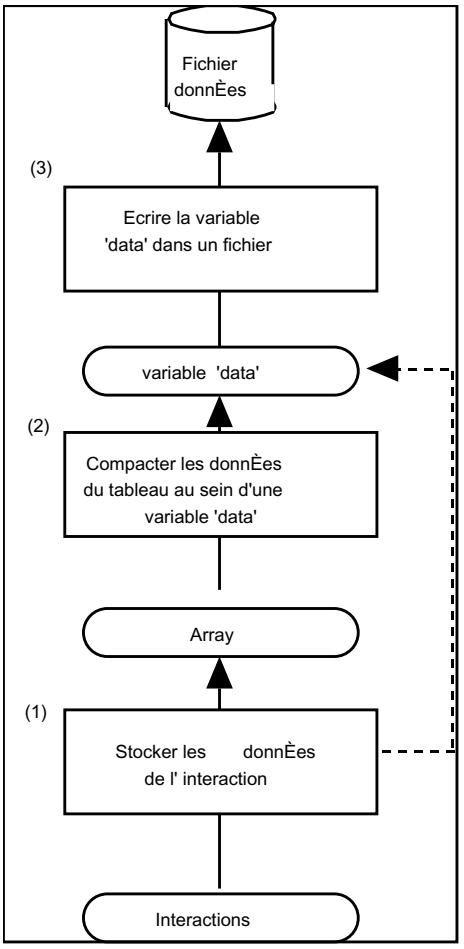


Figure 12-6:Algorithme de sauvegarde des données

L'étape (2) consiste à mettre bout-à-bout toutes les données dans une variable. Considérons par exemple, un tableau de 25 X 3 cases, chaque ligne contenant le numéro de la question , la réponse et le temps de réponse. La procédure (2) consisterait à ajouter ces trois informations les unes après les autres dans la variable 'data', suivies d'un saut de ligne (RETURN):

```
repeat with line := 1 to 25
    data := data ^ ("Question :"^ ArrayGet (((line - 1) * 3) + 1)))
    data := data ^ ("Réponse :" ^ ArrayGet (((line - 1) * 3) + 2)))
    data := data ^ ("Temps : "  ^ ArrayGet (((line - 1) * 3) + 3)))
    data := data ^ return
end repeat
```

L'utilisation d'un tableau (étape 2) n'est pas indispensable. Vous pouvez au fur et à mesure des interactions ajouter les données dans une variable texte. Attention toutefois à la longueur maximale des variables.

Ensuite, l'étape (3) se limite à écrire cette variable dans un fichier, le premier argument est le nom du fichier et la seconde le nom de la variable contenant toutes les informations.

```
WriteExtFile ("Reponses", data)
```

Lorsque les données concernent un utilisateur particulier, celles-ci seront généralement sauvées dans un fichier dont le nom est construit à partir du nom de l'utilisateur. Imaginons que vous ayez stocké ce nom sous

la variable NomUtilisateur, la sauvegarde serait par exemple réalisée par:

```
WriteExtFile (NomUtilisateur ^ ".resultats", data)
```

Dans ce cas, les données de Hans-Ruedi Oppliger seront par exemple sauvées sous "Oppliger.resultats". Si vous travaillez sous Windows 3.1., attention au nombre maximal de caractères autorisés pour le nom d'un fichier.

Exercice 12-3: Données sur l'utilisateur

Copiez le programme 12.2 et sauvez-le sous 12.3. Pour chaque question, sauvez la réponse de l'utilisateur et son temps de réponse. Sauvez aussi le score total du sujet et son temps de réponse moyen. Placez ces information dans un fichier de type "H.Zeller.data".

13. Aide

a Exercice 1-2: : “Hachures” : Choisir des hachures de direction différente (option 'fills' du menu 'Window/Inspectors') pour les deux ellipses et choisir le mode 'transparent' pour celle que vous avez créée en second lieu (option 'modes' du menu 'Window/Inspectors'). Pour éviter que le texte soit entouré d'un cadre blanc, il faut également choisir le mode transparent pour le texte. Attention, pour les Inspectors soient utilisables, il faut être en train d’éditer une icône de présentation.

b Exercice 1-3: : “Drapeau français” : Comme Authorware fonctionne en mode calque, il ne peut effacer partiellement les objets d’une même icône. Aussi, les trois bandes du drapeau doivent être dessinées dans des icônes de présentation différentes. Il faut placer ces icônes immédiatement les unes en dessous des autres.

c Exercice 1-4: : “Les quatre formes” : Dans une même icône d’effacement, toutes les icônes sélectionnées sont effacées par le même effet. Pour varier les effets, chaque forme doit être effacée au moyen d’une icône d’effacement différente. Placez ces icônes immédiatement les unes sous les autres

d Exercice 2-2: : “Questions siamoises” : La seconde interaction doit être construite à l’intérieur de la première. Dans le feedback associé à la réponse ’j’ai fini’ , placez une icône ’sous-programme’, dans laquelle vous construirez la deuxième interaction.

e Exercice 2-3: : “Question négative” : Dans ce cas comme la question est formulée négativement “ cliquer sur ce qui est faux ”, la réponse à définir comme ‘correcte’ est l’affirmation erronée “ Tous les nombres pairs sont multiples de quatre ” et vice-versa la réponse incorrecte est de dire que “ Tous les multiples de quatre sont des nombres pairs. ”. Ce piège vous était tendu afin de bien distinguer ce qui est vrai ou faux en général et ce que vous demandez à Authorware de comptabiliser comme réponse correcte ou incorrecte. Notez qu’il est généralement déconseillé de formuler des questions négatives, car un apprenant peut bien connaître son sujet, mais se laisser piéger par la formulation de la question. La question mesure alors davantage ses compétences linguistiques que sa connaissance du domaine.

f Exercice 2-5: : “Questions emboîtées” : Construisez d’abord l’interaction concernant les trois temps du mode indicatif et regroupez cette interaction au sein d’un sous-programme. Faites ensuite de même pour le mode conditionnel. Ensuite, posez l’icône d’interaction correspondant à la question principale (choix du mode) et posez les deux sous-programmes comme feedback de cette interaction principale.

g Exercice 3-2: : “Niveaux d’intentionnalité” : Vous aurez besoin de 8 zones de réponses. Les quatre zones déjà créées pour l'exercice 3.1. restent utiles quand l'utilisateur clique une fois. Ensuite, vous devez créer quatre zones identiques aux quatre premières mais en choisissant le paramètre 'cursor in area' pour l'option 'match with'. Le plus simple pour cela est de copier les quatre zones sensible créées dans l'execrice 3.1. puis les coller et modifier ce paramètre. Attention, placez les nouvelles réponses à droite (dans l'algorithme d'analyse) des zone sensibles déjà créées (nous verrons plus tard pourquoi). Ensuite, modifiez l'option 'inverse area' des zones sensibles 'single click'.

h Exercice 3-3: : “Zones imbriquées” : Nous avons donc trois zones imbriquées: tout point de la zone 'Berne' appartient aussi à la zone 'Autour de Berne', qui appartiennent à leur tour à la zone 'Suisse', ces trois zones doivent être placées dans cet ordre dans l'icône d'interaction. En effet, si la zone Suisse était placée en premier lieu (à gauche des deux autres), tout click sur la carte, y compris sur Berne, satisferait ce premier test et donnerait donc lieu à un feedback négatif.

i Exercice 3-4: : “La perpendiculaire” : Créez une icône de présentation dans laquelle vous dessinez le segment B. Ajoutez la lettre B au bout du segment et appelez l'icône 'Segment B'. Faites de même pour le segment C. Ensuite placez une icône d'interaction dans laquelle vous dessinez A et inscrivez la question elle-même. Pour positionner correctement les différents segments les uns par rapport aux autres, faites simplement un RUN et double-cliquez sur le segment que vous désirez déplacer. Ensuite placez une icône de présentation à droite de l'icône d'interaction et choisissez le mode de réponse 'clickable object'. Lorsque vous éditez la première réponse, Authoware vous demande de cliquer sur un objet, cliquez sur B et Authoware vous confirmera votre sélection en affichant 'target object: Segment B'. Si la droite B ne figure pas à l'écran au moment où vous devez la sélectionner, faites un run ou double-cliquez sur cette icône, avant d'éditer la réponse. Faites ensuite de même pour le segment C

j Exercice 3-5: : “La perpendiculaire saisissable” : Faites un copier-coller du segment visible. Donnez au

nouveau segment une largeur de plusieurs pixels. Rendez-le invisible, soit en lui donnant la même couleur que le fond d'écran, soit en le mettant en blanc et en choisissant le mode 'transparent'. Glissez-le au-dessus du segment visible et passez-le en arrière plan (option 'send to back' du menu 'attributes').

k Exercice 3-7: : “Puzzle” : La situation est inversée par rapport à l'exercice 3.6: en effet, il y a une zone de destination semblable (le trou dans le puzzle) pour trois objets différents. Utiliser l'option 'snap to center' pour la bonne pièce et l'option 'put back' pour les autres pièces.

l Exercice 3-8: : “Venn multiple” : Comme pour les zones sensibles, la zone de destination la plus large (tout l'écran), recouvrant les autres zones, sera placée en dernier lieu dans l'algorithme d'interaction (la plus à droite), car si elle était en premier lieur (à gauche) tout objet placé dans le diagramme de Venn serait d'abord considéré comme placé dans la zone 'tout l'écran'.

m Exercice 4-1: : “Droites perpendiculaires (suite)” : Traitez les feedback dans l'ordre suivant: (1) segment correct, (2) nombre maximal d'essais atteint, (3) segment incorrect et (4) click en dehors des deux segments, dans la zone sensible couvrant tout l’écran.

n Exercice 4-2: : “Limitation du temps de réponse” : Vous avez besoin de deux réponses 'time limit': pour la limite 60 secondes, choisissez l'option 'restart for each try'; pour la limite 180 secondes, ne sélectionnez pas cette option.

o Exercice 4-3: : “Cible” : Il s'agit de créer quatre zones sensibles concentriques et de les placer sur l'icône d'interaction de la plus petite à la plus large.

p Exercice 4-4: : “Désignation d’objets” : Cette fois, ne créez pas de zone spécifique pour l'intersection comme nous l'avions fait dans le programme? Créez une zone pour chaque ensemble, mais sélectionnez l'option 'continue' pour une des deux zones de telle sorte que l'utilisateur reçoive les deux feedback s'il positione le point dans l'intersection.

q Exercice 5-2: : “Matching des majuscules et minuscules” : Désélectionnez l’option 'ignore: capitalization'.

r Exercice 5-3: : “Jokers dans les réponses texte” : Pour la réponse correcte, utilisez le pattern hydrog?ne pour accepter ‘hydrogène’ avec ou sans accent. Pour la réponse mal orthographiée, utilisez le pattern *drog?ne pour accepter toute réponse plus ou moins correcte quels que soient les premiers caractères. Pour traiter en un feedback toutes les autres réponses, créez un pattern composé uniquement du pattern *. Attention, ce pattern identifie TOUTES les réponses, y compris les bonnes. Il doit donc être placé en dernier lieu dans l'analyse de réponse, c'est-à-dire après (à droite) que les autres réponses aient été identifiées et traitées.

s Exercice 5-4: : “Ordre des mots dans les réponses texte” : Les deux premières réponses, ordre correct et ordre incorrect, sont définies par le même pattern comprenant les quatre mots (sans guillements), mais avec l'option 'Ignore: word order' respectivement off et on. Pour la troisième réponse (erreur sur les éléments), accepter toute autre réponse (pattern = *)

t Exercice 5-6: : “Patterns complexes” : Pour la bonne réponse, le pattern est 'Genève Lausanne Fribourg Neuchatel Sion Martigny' avec l'option 'match at least 3 words'. Pour les réponses incomplètes, le pattern est le même mais indiquez 'match at least 1 words'

u Exercice 5-8: : “Analyses négatives” : Identifiez en premier lieu (à gauche dans l'interaction' le pattern 'pas'. Associez à cette réponse le feedback "Veuillez répondre sans utiliser la forme négative" et faites le recommencer.

v Exercice 5-9: : “Traduction anglais-français” : Cet exercice n'a pas de solution idéale. N'essayez pas pendant cinquante ans.

w Exercice 5-10: : “Texte Lacunaire” : Authorware ne traite qu'une zone de réponse par interaction. Vous devez donc créer une icône d'interaction pour chacund es deux trous dans le texte.

x Exercice 5-11: : “Réponses ‘touche’” : Chaque feedback est un sous-programme comprenant l'affichage de la forme et une icône de temporisation réglée sur 0.5 ou 0,5 seconde (selon la convention de représentation des nombres décimaux utilisée par votre machine). Pour effacer le feedback, choisissez l'option 'erase before next entry'.

y Exercice 6-1: : “Phrases chronométrées” : Le numéro de la phrase correspond au nombre d'essais (variable 'tries'). Le nombre de mots se trouve dans la variable 'WordCount'. Le temps nécessaire pour la dernière réponse se trouve dans la variable ResponseTime, qu'il ne faut pas confondre avec la variable TimeInInteraction, laquelle contient le temps écoulé au total des essais à une même interaction.

z Exercice 6-3: : “Rectangle temporel” : Le feedback du bouton est une icône de calcul dans laquelle figure la fonction ‘box’. Utilisez la variable ResponseTime comme paramètre de la fonction: par exemple, box (1, 100, 100, 100 + (* ResponseTime 10), 200)

aa Exercice 6-4: : “Cercle variable” : Lorsque vous ne connaissez pas le nom ou la syntaxe d'une fonction, sélectionnez 'functions' dans le menu 'Window', parcourez les fonctions proposées et lisez leur documentation. Ces variables sont aussi documentées dans le mode d'emploi (papier).

ab Exercice 6-5: : “Mini-calculette” : Créez une variable, par exemple 'resultat', qui vaut 0 au départ, et placez-la dans l'icône d'inetraction en sélectionnant l'option 'update displayed variable' (dans 'Display'). Le feedback de chaque bouton de la calculette est une icône de calcul dans laquelle la valeur de la variable resultat est modifiée.

ac Exercice 6-6: : “Mini-calculette (suite)” : Les trois variables doivent être alignées à droite (option 'right align' du sous-menu 'alignment' du menu 'text'). Limitez le nombres de chiffres décimaux et faites afficher les zéros complémentaires.

ad Exercice 6-7: : “Message d’attente” : Utilisez une boucle semblable à celle présentée ci-dessus. A chaque passage dans le corps de la boucle, utilisez la variable-compteur dans l'instruction 'box' pour déterminer la largeur du rectangle à dessiner. Les autres paramètres sont constants. Le premier argument de la fonction box sera -1 afin que le rectangle soit dessiné en plein.

ae Exercice 6-8: : “Dessin de cercles” : En d'autres termes, pour chaque valeur verticale, il faut traiter toutes les valeurs horizontales.

af Exercice 6-9: : “Dessin d’une diagonale de carrés” : Dans ce cas, les objets pour la diagnonale sont ceux pour lesquels N1 = N2.

ag Exercice 6-10: : “Message d’attente (suite)” : Enregistrez dans une variable le temps precis avant le début du dessin ('SystemSeconds). Ensuite dans la boucle, comparez le temps actuel (utilisant toujours 'SystemSeconds') au temps de départ. Si la différence dépasse 5 secondes, utilisez 'exit repeat'.

ah Exercice 7-1: : “Message vocal” : Si votre machine ne dispose pas d’un microphone, vous trouverez un fichier qui correspond à cette consigne sous ftp://tecfa.unige.ch/pub/sounds/. Sélectionnez l’option ‘concurrent’ de telle sorte que, dès que le son commence, le bouton ‘continue’ apparaisse (icône de temporisation).

ai Exercice 7-5: : “Cible mouvante” : Choisissez l’option de synchronisation ‘perpetual’ associée à une conditon telle que ‘stop = 0’. Créez une interaction avec réponse de type ‘objet sensible’, l’objet sélectionnable étant celui qui se déplace. Comme feedback pour cette réponse, placez l’instruction ‘stop := 1’. Si l’utilisateur clique sur l’objet, la condition permettant de continuer l’animation sera alors invalidée et l’animation s’arrêtera.

aj Exercice 7-6: : “Véhicule sur dents de scie” : Créez une trajectoire aux bornes 1 et 10. Pour chaque bouton, le feedback change la valeur de la variable ‘position’ (+ 1 ou - 1) et comprend ensuite l’animation proprement dite. Le bouton de gauche est actif (‘active if true’) que si la variable position est supérieure à 0; celui de droite, si cette variable est inférieure à 10.

ak Exercice 7-8: : “Agenda” : Utilisez deux variables ‘jour’ et ‘heure’ pour positionner la marque respectivement sur l’axe horizontal et vertical. Utilisez la grille d’écran (menu ‘View’, option ‘grid’ et ‘snap to grid’) pour faciler le dessin de l’agenda et le positionnement du curseur au point ‘base’ (jour=1, heure=1) et au point ‘end’ (jour=7 et heure=8)

al Exercice 8-2: : “Cible mouvante itérée (suite)” : Si vous définissez le click sur l’objet comme une ‘correct answer’, le critère de sortie sera la valeur de la variable ‘TotalCorrect’.

am Exercice 8-3: : “Formes aléatoires” : Créez individuellement six formes géométriques, chaque forme sur une icône de présentation distincte. Ensuite, placez ces icônes à droite d’une icône de décision, au mode de sélection aléatoire sans répétition.

an Exercice 8-6: : “Zombie” : Créez une animation dans laquelle l’objet se déplace vers une destination sur une grille, en fonction des variables PositionHoriz et PositionVerti, dont vous générez les valeurs aléatoirement. Les bornes de la fonction random devraient être identiques aux valeurs extrêmes (‘base’ et ‘end’) utilisées pour définir la grille d’animation. Placez la génération aléatoire et l’animation dans une boucle.

ao Exercice 8-7: : “Addition aléatoire” : En d’autres termes, la valeur minimale du deuxième nombre doit garantir que la somme dépasse 100, c’est-à-dire est au moins égal à la différence entre le premier nombre

et 100.

ap Exercice 8-8: : “Mot aléatoire” : Pour générer une lettre au hasard, génerez d’abord un nombre au hasard entre 97 et 122 et ensuite utilisez la fonction char pour obtenir une lettre à partir de ce nombre. Ne concaténez la nouvelle lettre au mot que si elle ne s’y trouve pas encore (find(lettre,mot)=0). Répétez la boucle tant que le mot comprend moins de 5 caractères (CharCount(mot)<5)

aq Exercice 9-1: : “Douanier sympathique” : Vous devez donc traiter deux patterns, un pour la réponse "Suisse" et un pour toutes les autres réponses (*). Cherchez dans les fonctions de la catégorie 'character' celle qui vous permet de mettre en majuscule la première lettre d'une chaîne de caractères.

ar Exercice 9-2: : “Altitude du Mont-Blanc” : N'oubliez pas de choisir le mode de branchement 'continue' pour la réponse 'texte'. En réalité, cette réponse ne nécessite pas de feedback. Néanmoins, si vous placez une icône de présentation vide, Authorware interrompera l'exécution du programme et repassera en mode 'édition' en pensant que vous avez oublié de dessiner le contenu de cette icôme. Pour éviter cela, placez un élément invisible sur l'icône de présentation, tel qu'un petit point transparent dans un coin ou “--” dans une icône de calcul.

as Exercice 9-3: : “Trois nombres croissants” : La condition définissant la réponse correcte est 'NumEntry < NumEntry2 & Numentry2 < NumEntry3'. La conditon pour détecter que l'utilisateur n'a pas introduit tous les nombres est 'NumCount < 3'. Enfin, vous devrez prévoir quatre erreurs: 'NumEntry > NumEntry2', 'NumEntry2 > NumEntry3', 'NumEntry > NumEntry3', NumEntry>NumEntry2>NumEntry3. Dans ces quatre cas, utilisez les variables dans le texte du feedback.

at Exercice 9-4: : “Multiplication de décimètres” : Deux solutions. Soit vous utilisez la condition suivante pour experimer la réponse correcte: (NumEntry=30& Unité="dm")|(NumEntry=3& Unité="m")|(NumEntry=300 & Unité="cm"). Soit, vous ramenez toute réponse en cm aprés avoir identifié l'unité: If Unité = "dm" then reponse := Numentry / 10 etc...

au Exercice 9-5: : “Les (inévitables) tables de multiplication” : Pour le dessin des pions, utilisez le programme 6.8. Créez un modèle à partir de 6.8 puis utilisez-le dans 9.5.

av Exercice 9-6: : “Le pluriel des noms” : Pour sélectionner les mots, placez dans une icône de décision, 10 icônes de calcul. Chaque icône donne des valeurs aux variables suivantes: NomSingulier (utilisé dans la question), NomPluriel (utilisé dans l'analyse de réponse) et NumRegle (numero de la règle d'accord: 1,2,3 ou 4.). La réponse sera correcte si find(NomPluriel, EnryText) > 0. Si ce n'est pas le cas, quatre réponses conditionnelles affichent la règle spécifiée selon la valeur de NumRegle.

aw Exercice 10-1: : “Score” : Le feedback du bouton sera un sous-programme comprenant une îcone de présentation, pour afficher les deux variables 'TotalCorrect' et 'TotalWrong' et une icône de temporisation (bouton continue), de telle sorte que l'utilisateur ait le temps de lire les informations. Ce feedback sera associé à l'option d'effacement 'before next entry', de telle sorte que tous soit effacé lorsque l'utilisateur revient à la question en cours.

ax Exercice 11-3: : “Hypertexte 3, titres” : Créez les deux icônes intercalaires et placez-les parmi les autres pages. Editez le framework. Ajoutez à l'icône d'interaction deux nouveaux boutons associés à une icône de navigation. Choisissez le type de destination 'anywhere' et cliquez sur l'icône 'chapitre 1 ' ou 'chapitre 2'.

ay Exercice 11-5: : “Hypertext 5, table des matières” : La table des matières est un document dans lequel tous les éléments (ou presque) sont des textes sensibles, renvoyant directement au titre considéré.

az Exercice 12-1: : “Capitale” : Il s'agit du même problème que celui décrit ci-avant, mais pris dans l'autre sens: lors de la génération aléatoire, il faut prélever à la fois le nom du pays et le nom de la capitale; lors de l'analyse de réponse il faut vérifier au moyen d'une réponse de type conditionnelle si la réponse de l'utilisateur correspond à la capitale choisie.

ba Exercice 12-2: : “Quiz” : Il s'agit d'une type de réponse bouton, dans laquelle vous noterez dans une variable si le bouton 'vrai' ou le bouton 'faux' a été sélectionné. Ensuite, utilisez une réponse de type condition pour vérifier si la réponse de l'utilisateur correspond à l'information ('vrai'/'faux') stockée avec la question.